```
% Main module - Multi-sectoral System Analysis
% Developed by R. Villarroel Walker, 2008-2016

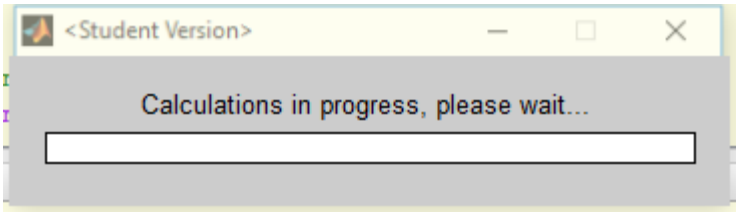
% deletes log file (a new one will be created later)
delete('msadiary.txt')

% Normal mode including multiroot
clearvars -except grand mat_sce
% Multiple RSA's
%clearvars -except tya para_zd_m para_pr_m para_un_m AAA
% Multiple attainability runs (pr)
%clearvars -except MAt rt tya
close all, format long, echo on
clc
%dbstop in hydpro01 at 172
grand = 1; % user: disable this line if multi-root is going to be used
wh = waitbar(0,'Calculations in progress, please wait...');
diary msadiary.txt
tic

% Define PATH of the model as 'upath'
upath = pwd; %ctfroot
% Define names of input/ouput files EXCEL
excelfile = 'MSAinputLON.xlsx'; %excelfile = char(filesname(1));
excelfile2 = 'MSAoutLON.xlsx';
```

```
Warning: File not found or permission denied
clc
%dbstop in hydpro01 at 172
grand = 1; % user: disable this line if multi-root is going to be used
wh = waitbar(0,'Calculations in progress, please wait...');
diary msadiary.txt
tic

% Define PATH of the model as 'upath'
upath = pwd; %ctfroot
% Define names of input/ouput files EXCEL
excelfile = 'MSAinputLON.xlsx'; %excelfile = char(filesname(1));
excelfile2 = 'MSAoutTEST.xlsx';
%%
%
%   for x = 1:10
%       disp(x)
%   end
%
% Definition of sub-paths
path0 = strcat(upath,'\wsector');
```



```
for x = 1:10
    disp(x)
end
```

```
% Definition of sub-paths
path0 = strcat(upath,'\wsector');
path1 = strcat(upath,'\fsector');
path2 = strcat(upath,'\esector');
path3 = strcat(upath,'\wmsector');
path5 = strcat(upath,'\general');
path6 = strcat(upath,'\rimmeb');
path7 = strcat(upath,'\sanalysis');
if ~isdeployed
    addpath(path0,path1,path2,path3,path5,path6,path7);
end
clearvars upath path0 path1 path2 path3 path5 path6 path7

% MODULE get input data ready for simulation
% Creates matrixes and specifies model running mode
dataready
waitbar(2/10,wh) % waiting bar

% DECISION TO RUN RIMME OR NOT
% if 'zero', parameters are assigned the expected value and RIMME code
% will be skipped. Model is run once using these parameter vectors
SumPara = sum(paramat(:,1)) + sum(paramat2(:,1)) + sum(initial(:,1)) + ...
    sum(input(size(input,1)-1,:),2);

if (modset(1) == 0) || (SumPara == 0);
    parause = paramat(:,2);
    parause2 = paramat2(:,2);
    statin = initial(:,2);
    inout = input(1:x,:);
    msa_main
end
% If 'one', RIMME algorithm is enabled. msa_main is run inside RIMME
% framework (see rsa_main).
waitbar(4/10,wh) % waiting bar

if (modset(1) == 1) && (SumPara > 0);
    rimmeb;

    %**************************************************************************
    % CHECKING ERROR LOG DURING RSA
    %**************************************************************************
    sumError = [(1:modset(6))' zeros(modset(6),1)];    % 5 major flows x 5[mean, SD]
    sumError(:,2) = sum(errorsRSA,2);
    save('sumError','sumError');
end

% Export some results to the export file (EXCEL)
waitbar(6/10,wh) % waiting bar
if modset(9) == 1
    Export
end

% computation here %
waitbar(10/10,wh)
pause(2)
diary off
```

```
close(wh)
toc
```

```
path1 = strcat(upath,'\fsector');
path2 = strcat(upath,'\esector');
path3 = strcat(upath,'\wmsector');
path5 = strcat(upath,'\general');
path6 = strcat(upath,'\rimmeb');
path7 = strcat(upath,'\sanalysis');
if ~isdeployed
    addpath(path0,path1,path2,path3,path5,path6,path7);
end
clearvars upath path0 path1 path2 path3 path5 path6 path7

% MODULE get input data ready for simulation
% Creates matrixes and specifies model running mode
dataready

nindi = 9;          % number of indicators
tsc   = 40;         % tsc is the number of state conditions (statout)
nflow = 355;        % number of flows (flowmat): originally 343
nsect = 5;          % number fo sectors (water, forestry, food, energy, and waste management)
nmat  = 7;          % number of materials/substances
% LIST OF SUBSTANCES AND UNITS
% ** 1 is water in Mgal/d (in calculations) but later converted to m3/s
% ** 2 is Nitrogen Metric tonne/year
% ** 3 is Phosphorus Metric tonne/year
% ** 4 is Carbon Metric tonne/year
% ** 5 is Energy in GWh/y
% ** 6 is Material flow in Metric tonne/year
% ** 7 is water in Metric tonne/year
% Multi scenario
if grand > 1
else
% In this step we extract data from the input file.
% model running parameters or setup
modset = xlsread(excelfile, 'readme');

% Definition of some global variables:
% Number of years of the simulation
x = modset(3);
% Disabling or enabling error dialog boxes (0 and 1 respectively)
edialog = modset(4);

% Error matrix
emat    = zeros(modset(6),1);

% Model parameters (some, not all)
para00 = xlsread(excelfile, 'paraTe');
para01 = xlsread(excelfile, 'paraWa');
para02 = xlsread(excelfile, 'paraFo');
para03 = xlsread(excelfile, 'paraFr');
para04 = xlsread(excelfile, 'paraEn');
para05 = xlsread(excelfile, 'paraWh');

% Concatenate parameters
paramat = [para00; para01; para02; para03; para04; para05];
clearvars para00 para01 para02 para03 para04 para05
% Sort based on the ID number (can be changed as all equations make use of
```

```
% this ID number
paramat = sortrows(paramat);
% Eliminate first three columns (not used by the model)
paramat(:,1:3) = [];

% MSA loads additonal parameters
% Initial conditions
initial = xlsread(excelfile, 'statin');
initial(:,1:3) = [];

% Composition of some flows (for mass/energy balance purposes)
paramat2 = xlsread(excelfile, 'content');
paramat2(:,1:3) = [];

% Energy profile of water flows (for energy balance purposes)
watEmat = xlsread(excelfile, 'waterE');

% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. Interacts with fe_mix
% rows = [IF SS PL CM SM CP] x columns = [grass forestry crops]
Data_mix = xlsread(excelfile, 'Fmix');
% Procedure for separating 'Fmix' data
t_1 = find(~isnan(Data_mix(:,1)));
t_2 = t_1;
t_2(1)=[]; t_2 =[t_2;1000];
t_3 = t_1(t_2 - t_1 > 1);

% Mix of area proportion of each fertilizer type
a_mix = Data_mix(1:t_3(1),:);
% Crop for Feed data (moisture, area proportion, yield)
% Also, any row with NaN is deleted
cr_mix = Data_mix(t_3(1)+1:t_3(2),:);
cr_mix(any(isnan(cr_mix),2),:) = [];
% Crop for Food data (moisture, area proportion, yield)
% Fruits
fr_mix = Data_mix(t_3(2)+1:t_3(3),:);
fr_mix(any(isnan(fr_mix),2),:) = [];
% Cereals
ce_mix = Data_mix(t_3(3)+1:t_3(4),:);
ce_mix(any(isnan(ce_mix),2),:) = [];
% Veggies
ve_mix = Data_mix(t_3(4)+1:t_3(5),:);
ve_mix(any(isnan(ve_mix),2),:) = [];

%%
% Base Case Values for RSA simulation
% Only if RSA is requested in addition to LHS, i.e. modset(19) == 0
if modset(19) == 0
    quack = xlsread(excelfile, 'RSA');
    base_v = quack(:,1); % Only column one
    base_v(isnan(base_v)) = []; % NaN eliminated
    % base_v must have a length equal to the numbers of years
    if length(base_v) > x
        base_v = base_v(1:x); % array is truncated to the number of years
    end
    if length(base_v) < x
    end
    % base_ID = identification # of indicators: [#, # within species, species]
    base_ID = quack(:,5:7);
    % delete any row with NaN
    base_ID(any(isnan(base_ID),2),:) = [];
```

```
    % If the user decides to explicitly define the behavior (target), that
    % means modset(18) equals 0. If MSA calculates the behavior
    % (improvement) then this is calculated in 'behdef1.m'
    if modset(18) == 0
        obje_v2 = quack(:,2);
        obje_v2(isnan(obje_v2)) = []; % NaN eliminated
        % For use with multipr.m user
        %obje_v2 = base_v * rt(tya);
        % obje_v2 must have a length equal to the numbers of years
        if length(obje_v2) > x
        obje_v2 = obje_v2(1:x); % array is truncated to the number of years
        end
        if length(obje_v2) < x
    end
end
%%
% Time series of inputs
input = xlsread(excelfile, 'inout');
len = size(input,2);
% Clean 'NaN' rows
input(1,:) = [];
input(any(isnan(input),2),:) = [];
% We correct matrix 'input' to match the number of years.
% Check excel file 'tab:inout' for number of rows of non-data (currently 3)
rc = size(input,1) - 3;

% If user enters more data than the number of years
% data in 'tab:inout' exceeds the needed for simulation
if rc(1) > x
    end
end
% If user enters less data than the number of years
% data in 'inout' tab is less than the needed for simulation
if rc(1) < x
end
% If the user introduces a zero for precipitation and watershed drainage
% (columns 1 and 2) the model estimates values similarly to the previous
if any(input(1:x,1) == 0) || any(input(1:x,2) == 0);
end
%%
% Check that a valid limiting nutrient has been selected (by the user) for
% land application. N or P only.
if (modset(7) == 1 || modset(7) == 2) == 0
    end
end
save mat_dat modset emat paramat initial paramat2 watEmat a_mix cr_mix ...
    fr_mix ce_mix ve_mix base_v base_ID input rc len

end % end of grand
% Pre-allocate matrix space
% statout - Output of state conditions
statout = zeros(x,tsc);
% Definition of matrices of flow and substances:
flowmat = zeros(x,nflow,nmat);
% definition of matrix containing calculated indicators
indimat = zeros(x,nmat,nindi);
% Category vectors (year, material, sector)
pro_vec = zeros(x,nmat,nsect);        % matrix of products
res_vec = zeros(x,nmat,nsect);        % matrix of inputs
```

```
was_vec = zeros(x,nmat,nsect);        % matrix of wastes
ae1_vec = zeros(x,nmat,nsect);        % matrix of air emissions
ae1_out = zeros(x,nmat,nsect);        % matrix of air emissions (outbound)
we1_vec = zeros(x,nmat,nsect);        % matrix of aquatic emissions
% Storage term (stock)
store = zeros(x,nmat,nsect);
% Desirable Emission levels (year, material)
ae0 = zeros(x,nmat);  % Healthy air emissions
we0 = zeros(x,nmat);  % Healthy aquatic emissions
waitbar(2/10,wh) % waiting bar

% DECISION TO RUN RIMME OR NOT
% if 'zero', parameters are assigned the expected value and RIMME code
% will be skipped. Model is run once using these parameter vectors
SumPara = sum(paramat(:,1)) + sum(paramat2(:,1)) + sum(initial(:,1)) + ...
    sum(input(size(input,1)-1,:),2);

if (modset(1) == 0) || (SumPara == 0);
end
% If 'one', RIMME algorithm is enabled. msa_main is run inside RIMME
% framework (see rsa_main).
waitbar(4/10,wh) % waiting bar

if (modset(1) == 1) && (SumPara > 0);
    rimmeb;

if modset(11) == 1
end
% parameter domain vectors
% RSA yes = 1, most likely, lower, upper, known uncertainty 1, 1
% define matrix [# of input variables] x [LHS sampling,likely,lower,upper,U
% calculated,U level] x [years]
% First we acommodate inputs (inout data from input file)in the parameters
% format with the following assumptions:
inputpar = [input(x+1,:)' input(1,:)' zeros(len,1) zeros(len,1) input(x+2,:)' input(x+3,:)'];
if x > 1
    end
end
%paracat = cat(1,paramat,paramat2,initial,inputpar);
paracat = cat(1,paramat,paramat2,initial,inputpar);

% indexing 1st parameter group (regular parameters)
pg1 = 1:size(paramat,1);
% indexing 2nd parameter group (content)
pg2 = size(paramat,1)+1:size(paramat,1)+ size(paramat2,1);
% indexing 3rd parameter group (initial values)
pg3 = pg2(size(pg2,2))+1:size(initial,1) + pg2(size(pg2,2));
% indexing 4th parameter group (inout, time changing characetristics)
pg4 = pg3(size(pg3,2))+1:size(inputpar,1) + pg3(size(pg3,2));

% Parameter vectors will be separated later, adapting to the structure of
% mainmsa.m  --> The last number of pgN should be same as size(paracat,1)
% Definition of uncertainty ranges based on schedule for those parameters
% indicated with unknown uncertainty
% Parameter indexes of those than uncertainty will be calculated
uc = find(paracat(:,5) == 1);
% Uncertainty level assigned for each parameter
uclevel = paracat(uc,6);
% Uncertainty factors
ucfactors = modset(size(modset,1)-4:size(modset,1),2);
% Array of uncertainty factors for each parameter
```

```
uncertmat = ucfactors(uclevel);
% Calculation of uncertainty range (upper and lower bound)
paracat(uc,3) = paracat(uc,2)./uncertmat; % Lower
paracat(uc,4) = paracat(uc,2).*uncertmat; % Upper

%% delete
%uncertmat = ones(size(paracat,1),1) * modset(size(modset,1)-4:size(modset,1),2)';
%paracat(uc,3) = paracat(uc,2)./uncertmat(:,paracat(uc,6));   % upper bound
%paracat(uc,4) = paracat(uc,2).*uncertmat(:,paracat(uc,6));   % lower bound
%uncertmat2 = ones(size(uc,1),1) * modset(size(modset,1)-4:size(modset,1),2)';
%%
% If the user prefers to use parameter vectors from previous samplings,
% i.e. modset(11) = 1, then 'par_latin' is disabled and a parameter vector
% must have been loaded in line 18.
if modset(11) == 0
    par_latin

npoint = modset(5);             % # parameter vectors (default 5000).
                               % Adjusted in the model setting spreadsheet
nintvl = 20;                                    % # intervals for LHS (default (20)

to_sample = find(paracat(:,1)==1);      % index of parameters to be sampled

to_fix    = find(paracat(:,1)==0);      % index of parameters to be fixed

npar      = size(to_sample,1);          % # parameters to be sampled

% Redefine lower and upper bounds as row vectors.
dL = (paracat(to_sample,3))'; dU = (paracat(to_sample,4))';

% Build matrix of trial parameter vectors using
% the LHS technique.
% 'abso'  = absolute values
% 'norm' = rescaled values [0,1]
[abso,norm] = latin(npoint,npar,nintvl,dL,dU);

clear dL dU nintvl npar npoint

save msalatin abso norm to_sample to_fix -double
end
rsa_main

nsim = size(abso,1);            % #(Monte Carlo simulations)
npar = size(abso,2);            % # sampled model parameters

% Run the model, return the simulation trajectories, and
% test behaviour/nonbehaviour for each parameter vector.
% Pre-allocate matrix space
beh = zeros(x,nsim);

infbodRSA  = zeros(x,nsim);
effbodRSA  = zeros(x,nsim);
indimatRSA  = zeros(x,nmat,nindi,nsim);
flowmatRSA  = zeros(x,nflow,nmat,nsim);
errorsRSA   = zeros(modset(6),nsim);
statoutRSA  = zeros(x,tsc,nsim);
% five types of major flows: products, reasources, waste, air emissions,
% and water emissions
mflowRSA  = zeros(x,nmat,5,nsim);
% Values for healthy emissions
```

```
airRSA = zeros(x,nmat,nsim);
watRSA = zeros(x,nmat,nsim);

for sa=1:nsim

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%*********************************************************************
% Store data
%*********************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
```

```
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%************************************************************************
% Estimations associated with land uses
%************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%************************************************************************
% ESTIMATION OF AREA for each designated land use
%************************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
```

```
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
```

```
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
```

```
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%************************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%************************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% ************************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ************************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
```

```
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% ************************************************************************
% CALCULATION OF food consumption (by humans)
% ************************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
```

```
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
```

```
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);
```

```
%************************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%************************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%************************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%************************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
```

```
end

%*********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%*********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
```

```
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%**********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%**********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
```

```
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
```

```
end
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%**********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%**********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
```

```
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
```

```
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                    % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);              % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%*********************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%*********************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
```

```
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));
```

```
        % Estimation of Evapotranspiration for each surface in mm/year
        % [water forest grass crops -impervious]
        Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


        % Limit evaporation to the amount of precipitation for
        % [forest grass crops] in mm/year
        for luse_et = 2:4
           if Q_Et(luse_et,4) > precip
               Q_Et(luse_et,4) = precip;
           if Q_Et(luse_et,4) > precip
           if Q_Et(luse_et,4) > precip
           end
        end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);
```

```
% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
```

```
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    end
    % (b) sewer infiltration index
    sinfil = parause(55) * sinfil;

    % Runoff from pervious areas
    flowmat(:,77,:) = epa - ginfil;

    % Calculation of the fertilization mix (area-wise)
    fertmix

    tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
    tpa = statout(:,27); % total pervious area km2
    %tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
    Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
    % Forested areas accounts also for urban forests
    %*********************************************************************
    % FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
    %*********************************************************************
    % Creates a matrix of the proportion of non-fertilized area for each land
    % use with the size of the alternatives of fertilization
    fractnF = zeros(tfe,size(Areas,2));
    % [grass forestry crops]
    for a = 1:tfe;
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
    end

    % Specification of the fertilizer mix (in terms of area proportion) for
    % each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
    fe_mix = a_mix.* (ones(tfe,3) - fractnF);
    % fe_mix modified. non-fertilized area fraction is added.
    % fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
    fe_mix = cat(1,fractnF(1,:),fe_mix);
    % depth = [number of years]
    fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
    for a = 1:x % each year
        for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
```

```
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%************************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
```

```
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%*********************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%*********************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%*********************************************************************
% WATER OUTPUT CALCULATIONS
%*********************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%*********************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%*********************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
```

```
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
```

```
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%*********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%*********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
```

```
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%*********************************************************************
% Algae growing from WWTP effluent
%*********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
```

```
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
```

```
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%**********************************************************************
% Emissions to air
%**********************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%**********************************************************************
% Activated sludge reactor
%**********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
```

```
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%************************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%************************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ====================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
```

```
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%************************************************************************
% BIOGAS FLOW
%************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%************************************************************************
% TREATED SLUDGE FLOW
%************************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
```

```
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*****************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*****************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
```

```
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]
```

```
% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ************************************************************
% Calculation of live tree removals
% ************************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%*********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
```

```
%*********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%*********************************************************************
% CALCULATION OF CARBON POOLS
%*********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
```

```
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
        if a == 1
            statout(a,34) = smcap1(a) + smcap0;
        end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
```

```
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*****************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*****************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*****************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*****************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*****************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*****************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;
```

```
% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
%*********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%*********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%*********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
```

```
        flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
        man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
        flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
        flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
        flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
        b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
        flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
        man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
        flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
        flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
        flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    end
    % manure lbs per head per day (crosschecking)
    % for poultry
    % (flowmat(:,310,6)./statout(:,6))/365 * 2.2
    % Energy content calculated based on Channiwala 2002
    % HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
    % HHV in MJ/kg of dry mass
    % Estimation for H S O Ash is based on their relationship to carbon content
    % in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
    % and A/C = 0.6783
    % Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
    % check O/C = 0.95 approximately
    UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
    Mener   = zeros(3,1); % [cattle poultry swine]
    for a = 0:5:10
        b = a/5;
        Car = parause2(111+a);
        UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
        UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

        % HHV in MJ/kg of dry mass
        Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
            + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
            - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
        % Energy in kWh/y
        flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
        b = a/5;
        Car = parause2(111+a);
        UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
        UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

        % HHV in MJ/kg of dry mass
        Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
            + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
            - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
        % Energy in kWh/y
        flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
        b = a/5;
        Car = parause2(111+a);
        UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
        UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

        % HHV in MJ/kg of dry mass
        Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
            + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
            - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
        % Energy in kWh/y
        flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    end
```

```
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%**********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%**********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
```

```
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%*************************************************************
% Recommended Nutrient FOR EACH LAND USE
%*************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
```

```
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
     end
     bB = bB + b + 1;
     for b = 1:3 % [N P C]
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
     end
     bB = bB + b + 1;
 end
 %****************************************************************
 % Nutrient Content for each type of fertilizer IF SS PL CM SM CP
 %****************************************************************
 % First we calculate the water content (moisture) of manure after storage
 % & handling (estimated in manure.m) as a fraction w/w of total mass,
 % nutrient content as w/w fraction of DM, and energy in kWh/tonne
 fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

 % Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
 % Column of zeros for IF
 for a = 1:nmat
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
 end
 % Now data is corrected to reflect fraction of mass. For compost we use the
 % typical compost composition entered by user
 % Dry matter in tonne/y
 fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
 % We convert first column from Water content Mgal/d to w/w of total mass
 fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

 % Nutrient content corrected to w/w of dry matter
 for a = 2:4
     fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
     fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
     fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
 end
 % Energy content corrected to kWh per tonne (fresh matter)
 fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

 % fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
 % fraction as from 1 to 5 are.
 % Data is added for IF (not calculated above) from user input
 % Assumed moisture 0.5%
 fm_cont(isnan(fm_cont)) = 0;
 fm_cont(:,1,1) = ones(x,1) * 0.005;
 fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;
```

```
% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
```

```
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
```

```
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);
```

```
% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%*************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%*************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%******************************************************************
% MANURE distribution
%******************************************************************
% Calculates de amount of manure sent to waste management and the manure
```

```
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%************************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%************************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%*********************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
```

```
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%*********************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%*********************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%*********************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%*********************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%*********************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
```

```
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
```

```
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%**************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%**************************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
```

```
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
```

```
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%**************************************************************************
% CONVERSIONS
%**************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
```

```
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%*********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
```

```
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
```

```
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
```

```
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  ************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  ************************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
```

```
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
end
% Total meat produced
flowmat(:,160,:) = sum(P_meat,2);

% Total animal (meat) mass accumulated due to increase or decrease of
% livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
L_acc = zeros(x,3,nmat);
% Average livestock growth in tonne/y, based on growth of thousand
% heads: 'liveg' calculated in 'livepop.m'
for b = 0:2
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
end
% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
end
% Total livestock accumulation
flowmat(:,306,:) = sum(L_acc,2);

% DAIRY  ***********************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
```

```
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);            % tonnes/y
```

```
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);           % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ***********************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  ************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*******************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*******************************************************************
```

```
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively
```

```
% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*********************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*********************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);
```

```
% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
```

```
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%*********************************************************************
% FOOD SLUDGE FLOW
%*********************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%*********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%*********************************************************************
mswinci
```

```
AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%***********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%***********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
```

```
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%*********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%*********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%*********************************************************************
% First part of the energy module. second part is 'energys2.m'
%*********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
```

```
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%**************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%**************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%**************************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%**************************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
```

```
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%*********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
```

```
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%*********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%*********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
```

```
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                           % content in raw coal (dry basis)
                           % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%**************************************************************************
% Emissions from power generation (coal)
%**************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%**************************************************************************
% Algae growing from coal plant emissions
%**************************************************************************
```

```
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%**************************************************************************
% Production of biodiesel
%**************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
```

```
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%**************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%**************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;
```

```
% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%**********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%**********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**********************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%**********************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%**********************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%**********************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
      m_H_pl = 0;
      m_O_pl = 0;
```

```
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%**************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%**************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
```

```
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
```

```
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);
```

```
% Checking: sum(pyFlows(:,1:8,1),2)
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                    % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                    % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                    % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%*********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%*********************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%*********************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%*********************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);
```

```
%************************************************************************
% LANDFILLING
%************************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
```

```
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
```

```
end
%*********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%*********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%*********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%*********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
```

```
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    end
    % Resulting composted fertilizer - CORRECTING TOTAL MASS
    % Substracting water and nutrient loss
    flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
    % check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
    % Energy content calculated based on Channiwala 2002
    % HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
    % HHV in MJ/kg of dry mass
    % Estimation for H S O Ash is based on their relationship to carbon content
    % in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
    % and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
    % compared to fresh manure ~20-30%. So we double of Singh's values,
    % Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
    % with the ~51% reduction in Carbon content during composting that promotes
    % a larger Ash/C ratio.
    % Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
    % check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
    UAn     = zeros(x,6);    % percentage [C H S O N Ash]
    Drym    = flowmat(:,315,6) - flowmat(:,315,7);
    UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
    UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
    UAn(:,5) = flowmat(:,315,2)./Drym * 100;
    UAn(:,6) = UAn(:,1) * 1.4;
    UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

    %Energy in MJ/kg
    comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
            + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
            - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);
```

```
% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%*****************************************************************
% EXPORTS/IMPORTS
%*****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%*****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%*****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
```

```
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%*********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
```

```
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%*********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%*********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
```

```
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);
```

```
% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
```

```
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
```

```
%************************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%************************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%************************************************************************
% Water
%************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%************************************************************************
% Nitrogen
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
```

```
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%-----------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%-----------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
```

```
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%************************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%************************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%************************************************************************
% Forestry Sector
%************************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
```

```
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%*********************************************************************
% Food Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%*********************************************************************
```

```
% Energy Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%*********************************************************************
% Waste Management Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
```

```
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%************************************************************************
% INDICATOR SCORES ARE CALCULATED                                       *
%************************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%************************************************************************
%% NEW indicator
%************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
```

```
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
```

```
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
```

```
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
```

```
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%************************************************************************
% Store data
%************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%************************************************************************
% Estimations associated with land uses
%************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
```

```
    emat(2) = 0;
end
%*********************************************************************
% ESTIMATION OF AREA for each designated land use
%*********************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
```

```
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
```

```
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%*********************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%*********************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
```

```
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% ************************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ************************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% ************************************************************************
% CALCULATION OF food consumption (by humans)
% ************************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
```

```
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
```

```
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
```

```
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
```

```
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%**********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%**********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
```

```
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%*********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%*********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
```

```
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
```

```
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
end
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%*********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%*********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
```

```
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;
```

```
% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                   % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);             % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%*********************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
```

```
%************************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
```

```
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
```

```
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
```

```
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
end
% (b) sewer infiltration index
sinfil = parause(55) * sinfil;

% Runoff from pervious areas
flowmat(:,77,:) = epa - ginfil;

% Calculation of the fertilization mix (area-wise)
fertmix

tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
tpa = statout(:,27); % total pervious area km2
%tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
% Forested areas accounts also for urban forests
%*********************************************************************
% FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
%*********************************************************************
% Creates a matrix of the proportion of non-fertilized area for each land
% use with the size of the alternatives of fertilization
fractnF = zeros(tfe,size(Areas,2));
% [grass forestry crops]
for a = 1:tfe;
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
end

% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
```

```
fe_mix = a_mix.* (ones(tfe,3) - fractnF);
% fe_mix modified. non-fertilized area fraction is added.
% fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = cat(1,fractnF(1,:),fe_mix);
% depth = [number of years]
fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
for a = 1:x % each year
    for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%**************************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%**************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
```

```
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%************************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%************************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%************************************************************************
% WATER OUTPUT CALCULATIONS
%************************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%************************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
```

```
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%**************************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
```

```
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%*********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%*********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
```

```
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
```

```
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%**********************************************************************
% Algae growing from WWTP effluent
%**********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
```

```
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%**************************************************************************
% Emissions to air
%**************************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
```

```
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%*********************************************************************
% Activated sludge reactor
%*********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%*********************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%*********************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
```

```
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%**************************************************************************
% BIOGAS FLOW
%**************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%**************************************************************************
% TREATED SLUDGE FLOW
%**************************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
```

```
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*******************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
```

```
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*******************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;
```

```
% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ********************************************************
% Calculation of live tree removals
% ********************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*****************************************************************
```

```
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%*********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%*********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%*********************************************************************
% CALCULATION OF CARBON POOLS
%*********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
```

```
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
```

```
        if a == 1
            statout(a,34) = smcap1(a) + smcap0;
        end
    end
    % Estimation of the maximum input capacity of sawmills based on the throuput
    % and the amount of byproducts (tonne/y). The flow of wood to sawmills is
    % assumed to be equal to their capacity.
    % input capacity includes wood and bark
    flowmat(:,101,6) = statout(:,34)./parause(74);
    % Nutrient and Energy content based on amount of bark and wood
    flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
        (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
    flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
    flowmat(:,101,1) = flowmat(:,101,7) /1382000;

    %*********************************************************************
    % CALCULATION OF SAWMILL OUTPUTS
    %*********************************************************************
    % Final product from sawmills to "Local Consumption" (tonne/y)
    flowmat(:,103,6) = statout(:,34);
    % Nutrient and Energy content based on wood composition only (no bark)
    flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
    flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
    flowmat(:,103,1) = flowmat(:,103,7) /1382000;

    %*********************************************************************
    % CALCULATION OF SAWMILL RESIDUE
    %*********************************************************************
    % Residue of sawmills
    flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
    %flowmat(:,101,2)./flowmat(:,101,6)
    %flowmat(:,103,2)./flowmat(:,103,6)
    % Sawmill residue not used, i.e. exported (negative value)
    flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
    % Sawmill residue used as industrial fuel
    flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
    % Sawmill residue used as a product in the local market
    flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
    %flowmat(:,102,2)./flowmat(:,102,6)
    %*********************************************************************
    % CALCULATION OF WOOD for PAPER PRODUCTION
    %*********************************************************************
    % Paper production and composition
    flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
    % Nutrient and Energy content based on paper composition
    flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
    flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
    flowmat(:,327,1) = flowmat(:,327,7) /1382000;

    % Input of wood for paper production - a function of the pulp extraction
    % process as described by parause(73)
    flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
    % Nutrient and Energy content based on wood composition only (no bark)
    flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
    flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
    flowmat(:,328,1) = flowmat(:,328,7) /1382000;

    %*********************************************************************
    % CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
    %*********************************************************************
    % Estimation of the annual internal use of wood products based on
```

```
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
%*********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
```

```
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%*********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%*********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));
```

```
    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%**********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%**********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
```

```
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
```

```
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%*****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%*****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
end
%*****************************************************************
% Nutrient Content for each type of fertilizer IF SS PL CM SM CP
%*****************************************************************
% First we calculate the water content (moisture) of manure after storage
% & handling (estimated in manure.m) as a fraction w/w of total mass,
% nutrient content as w/w fraction of DM, and energy in kWh/tonne
fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

% Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
% Column of zeros for IF
for a = 1:nmat
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
end
% Now data is corrected to reflect fraction of mass. For compost we use the
% typical compost composition entered by user
% Dry matter in tonne/y
```

```
fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
% We convert first column from Water content Mgal/d to w/w of total mass
fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

% Nutrient content corrected to w/w of dry matter
for a = 2:4
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
end
% Energy content corrected to kWh per tonne (fresh matter)
fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

% fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
% fraction as from 1 to 5 are.
% Data is added for IF (not calculated above) from user input
% Assumed moisture 0.5%
fm_cont(isnan(fm_cont)) = 0;
fm_cont(:,1,1) = ones(x,1) * 0.005;
fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
```

```
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
```

```
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
```

```
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%*************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%*************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
```

```
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%*********************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%*********************************************************************
```

```
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%**************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%**************************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%**************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%**************************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%**************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%**************************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%**************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%**************************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching
```

```
floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
```

```
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%*********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
```

```
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%**************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
```

```
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%*********************************************************************

 %*********************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%*********************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%*********************************************************************
% CONVERSIONS
%*********************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
```

```
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%**********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
```

```
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
```

```
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
```

```
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  ************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  ************************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end
```

```
% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
end
% Total meat produced
flowmat(:,160,:) = sum(P_meat,2);

% Total animal (meat) mass accumulated due to increase or decrease of
% livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
L_acc = zeros(x,3,nmat);
% Average livestock growth in tonne/y, based on growth of thousand
% heads: 'liveg' calculated in 'livepop.m'
for b = 0:2
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
end
% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
```

```
end
% Total livestock accumulation
flowmat(:,306,:) = sum(L_acc,2);

% DAIRY  ************************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
```

```
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);            % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);            % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ***********************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y
```

```
% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  ************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
```

```
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*********************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*********************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
```

```
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig
```

```
FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%**************************************************************************
% BIOGAS FLOW
%**************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%**************************************************************************
% FOOD SLUDGE FLOW
%**************************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
```

```
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%*********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%*********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
```

```
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%**********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%**********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%**********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%**********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%**********************************************************************
% First part of the energy module. second part is 'energys2.m'
%**********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;
```

```
% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%**************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%**************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
```

```
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%*********************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%*********************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%*********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
```

```
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%**********************************************************************
% Estimation of flows for power generation
%**********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%**********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%**********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
```

```
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                            % content in raw coal (dry basis)
                            % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
```

```
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%************************************************************************
% Algae growing from coal plant emissions
%************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%************************************************************************
% Production of biodiesel
%************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
```

```
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%**************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%**************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
```

```
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%**********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%**********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**********************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
```

```
%************************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%************************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%************************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
```

```
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
```

```
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
   pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
   pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
```

```
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

% Checking: sum(pyFlows(:,1:8,1),2)
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                    % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                    % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                    % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
```

```
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%**************************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%**************************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%**************************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%**************************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%**************************************************************************
% LANDFILLING
%**************************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
```

```
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year
```

```
% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%**********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%**********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%**********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%**********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
```

```
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
end
% Resulting composted fertilizer - CORRECTING TOTAL MASS
% Substracting water and nutrient loss
flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
% check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
```

```
% and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
% compared to fresh manure ~20-30%. So we double of Singh's values,
% Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
% with the ~51% reduction in Carbon content during composting that promotes
% a larger Ash/C ratio.
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
UAn     = zeros(x,6);    % percentage [C H S O N Ash]
Drym    = flowmat(:,315,6) - flowmat(:,315,7);
UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = flowmat(:,315,2)./Drym * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%****************************************************************
% EXPORTS/IMPORTS
%****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
```

```
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
```

```
% Includes calculation of imports or exports
%*********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
```

```
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%**********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%**********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF      % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
```

```
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');
```

```
    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
```

```
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%*********************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%*********************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%*************************************************************************
% Water
%*************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%*************************************************************************
% Nitrogen
%*************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
```

```
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
```

```
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%************************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%************************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);
```

```
% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%*********************************************************************
% Forestry Sector
%*********************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%*********************************************************************
% Food Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
```

```
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%**********************************************************************
% Energy Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%**********************************************************************
% Waste Management Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
```

```
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%*************************************************************************
% INDICATOR SCORES ARE CALCULATED                                        *
%*************************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%*************************************************************************
%% NEW indicator
%*************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
```

```
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
```

```
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0
```

```
else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main
```

```
prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%*************************************************************************
% Store data
%*************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%*************************************************************************
% Estimations associated with land uses
%*************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);
```

```
% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%*********************************************************************
% ESTIMATION OF AREA for each designated land use
%*********************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
```

```
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
```

```
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
```

```
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%************************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%************************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% ************************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ************************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% ************************************************************************
% CALCULATION OF food consumption (by humans)
% ************************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
```

```
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
```

```
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);
```

```
%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
```

```
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%*********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%*********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
```

```
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
```

```
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%*****************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%*****************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
```

```
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    end
    % Correction for air cooling technology for natural gas power generation
    flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

    % Total water withdrawal GW + SW
    flowmat(:,59,1) = withtot;
    %Total water withdrawn GW
    flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
    for a = 1:x
        flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
    end
    % Total water withdrawal SW
    flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
    for a = 1:x
        flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
    end
    % Total water withdrawal GW + SW (COMPOSITION)
    flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

    % Total consumptive water use
    flowmat(:,60,1) = -1 * consutot;
    % Total water discharged from power generation processes
    flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);
```

```
%*********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%*********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
```

```
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01
```

```
imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                    % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);              % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%************************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%************************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
```

```
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
```

```
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
```

```
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
end
% (b) sewer infiltration index
sinfil = parause(55) * sinfil;

% Runoff from pervious areas
flowmat(:,77,:) = epa - ginfil;

% Calculation of the fertilization mix (area-wise)
fertmix

tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
tpa = statout(:,27); % total pervious area km2
%tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
```

```
% Forested areas accounts also for urban forests
%*********************************************************************
% FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
%*********************************************************************
% Creates a matrix of the proportion of non-fertilized area for each land
% use with the size of the alternatives of fertilization
fractnF = zeros(tfe,size(Areas,2));
% [grass forestry crops]
for a = 1:tfe;
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
end

% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = a_mix.* (ones(tfe,3) - fractnF);
% fe_mix modified. non-fertilized area fraction is added.
% fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = cat(1,fractnF(1,:),fe_mix);
% depth = [number of years]
fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
for a = 1:x % each year
    for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%***********************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%***********************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);
```

```
% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%*********************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%*********************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
```

```
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%*********************************************************************
% WATER OUTPUT CALCULATIONS
%*********************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%*********************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%*********************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
```

```
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
```

```
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%*********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%*********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
```

```
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%**********************************************************************
% Algae growing from WWTP effluent
%**********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW
```

```
if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%*********************************************************************
% Emissions to air
%*********************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;
```

```
% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%**************************************************************************
% Activated sludge reactor
%**************************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;
```

```
% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%*********************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%*********************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ===================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
```

```
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%*********************************************************************
% TREATED SLUDGE FLOW
%*********************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
```

```
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*****************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*****************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
```

```
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ***********************************************************
% Calculation of live tree removals
% ***********************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
```

```
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%*********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%*********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%*********************************************************************
% CALCULATION OF CARBON POOLS
%*********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));
```

```
% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
```

```
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
       if a == 1
           statout(a,34) = smcap1(a) + smcap0;
       end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
```

```
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);
```

```
% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
%*********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
```

```
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%*********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%*********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
```

```
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
```

```
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%*****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%*****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
end
%*****************************************************************
% Nutrient Content for each type of fertilizer IF SS PL CM SM CP
%*****************************************************************
% First we calculate the water content (moisture) of manure after storage
% & handling (estimated in manure.m) as a fraction w/w of total mass,
% nutrient content as w/w fraction of DM, and energy in kWh/tonne
fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

% Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
% Column of zeros for IF
```

```
for a = 1:nmat
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
end
% Now data is corrected to reflect fraction of mass. For compost we use the
% typical compost composition entered by user
% Dry matter in tonne/y
fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
% We convert first column from Water content Mgal/d to w/w of total mass
fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

% Nutrient content corrected to w/w of dry matter
for a = 2:4
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
end
% Energy content corrected to kWh per tonne (fresh matter)
fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

% fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
% fraction as from 1 to 5 are.
% Data is added for IF (not calculated above) from user input
% Assumed moisture 0.5%
fm_cont(isnan(fm_cont)) = 0;
fm_cont(:,1,1) = ones(x,1) * 0.005;
fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);
```

```
% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
```

```
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%**********************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
```

```
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%**************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%**************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
```

```
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
```

```
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%************************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%************************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%*********************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%*********************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%*********************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%*********************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%*********************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
```

```
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%*********************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
```

```
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
```

```
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%**********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
```

```
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
```

```
%************************************************************************
% CONVERSIONS
%************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%************************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%************************************************************************
```

```
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
```

```
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
```

```
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
```

```
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  *************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  *************************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
end
% Total meat produced
flowmat(:,160,:) = sum(P_meat,2);

% Total animal (meat) mass accumulated due to increase or decrease of
% livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
L_acc = zeros(x,3,nmat);
% Average livestock growth in tonne/y, based on growth of thousand
% heads: 'liveg' calculated in 'livepop.m'
for b = 0:2
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
end
% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
```

```
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
end
% Total livestock accumulation
flowmat(:,306,:) = sum(L_acc,2);

% DAIRY  ************************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
```

```
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);            % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
```

```
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;    % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);            % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ************************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;    % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);            % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  *************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
```

```
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*************************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*************************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
```

```
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
```

```
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%*********************************************************************
% FOOD SLUDGE FLOW
%*********************************************************************
% Calculated as the difference between feed and biogas
```

```
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%*********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%*********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
```

```
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%*********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%*********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%*********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%*********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%**********************************************************************
% First part of the energy module. second part is 'energys2.m'
%**********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
```

```
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
```

```
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%**********************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%**********************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%**********************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%**********************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%**********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%**********************************************************************
```

```
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%**********************************************************************
% Estimation of flows for power generation
%**********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%**********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%**********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
```

```
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                           % content in raw coal (dry basis)
                           % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
```

```
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%************************************************************************
% Algae growing from coal plant emissions
%************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;
```

```
% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%**********************************************************************
% Production of biodiesel
%**********************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%**********************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%**********************************************************************
```

```
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%*********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%*********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;
```

```
% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**********************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%**********************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%**********************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%**********************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
```

```
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
```

```
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
```

```
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

% Checking: sum(pyFlows(:,1:8,1),2)
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;        % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;        % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;        % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
```

```
    flowmat(:,265,a) = pyFlows(:,a,2);                  % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                  % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                  % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                    % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                    % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                    % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%**************************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%**************************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%**************************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%**************************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%**************************************************************************
% LANDFILLING
%**************************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);
```

```
% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
```

```
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%*********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%*********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%*********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%*********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
```

```
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
```

```
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
end
% Resulting composted fertilizer - CORRECTING TOTAL MASS
% Substracting water and nutrient loss
flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
% check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
% and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
% compared to fresh manure ~20-30%. So we double of Singh's values,
% Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
% with the ~51% reduction in Carbon content during composting that promotes
% a larger Ash/C ratio.
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
UAn     = zeros(x,6);    % percentage [C H S O N Ash]
Drym    = flowmat(:,315,6) - flowmat(:,315,7);
UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = flowmat(:,315,2)./Drym * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%*****************************************************************
% EXPORTS/IMPORTS
%*****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
```

```
    else
        emat(15) = 1;
    end
end
%*****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%*****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
```

```
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%*********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x
```

```
        % No local consumption of diesel
        if  flowmat(a,199,6) == 0
            end

        % There is local consumption of diesel
        else
            if impDI(a) < 0 && isnan(impDI(a))
                end
             else % impDI(a) > 0 or diesel has been used up for power
                % all diesel (fossil and bio) is imported
                flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
                % imported diesel (fossil)
                flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
                % imported diesel (bio)
                flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
                flowmat(a,247,:) = flowmat(a,241,:);
            end
        end
    end
    % Gasoline characteristics and import/export estimation
    flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
    flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
    % 240   Imports/exports of gasoline (fossil fuel only and no production)
    % Assumes also that the minimum content of biofuels is achieved (imported)
    flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
    % Import of bio-gasoline (includes biofuel for diesel engines)
    flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
    % Total biofuel used for engines (gasoline and diesel)
    flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

    %*********************************************************************
    % ESTIMATION OF EMISSIONS from local use and power
    %*********************************************************************
    % Emissions from fuel consumption in tonne/y (non power)
    emissionF       % by FUEL TYPE

    for a = 1:nmat
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
```

```
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    end
    % Biomass emission factor tonne/kwh (assumed as wood)
    bitkwh = parause2(129:131) * parause2(132)^-1;
    bitkwh(2) = 0; % Phosphorus corrected to be zero

    %%% Correction. Natural gas, gasoline, and diesel are assumed to
    %%% undergo 100% oxidation, i.e. all C and N is released as emission
    % Natural gas emission factor tonne/kwh (100% oxidation)
    %ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
    % Gasoline emission factor tonne/kwh (100% oxidation)
    %gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
    % Diesel emission factor tonne/kwh (100% oxidation)
    %ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
    % 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
    % Coal combustion generates CCP and emissions, partitioning input materials
    tol_coal_em = zeros(x,3);
    for a = 1:x
        tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
    end
    flowmat(:,219,2:4) = -1 * (tol_coal_em);

    % 220   Emissions from Local NG use (DO, CO, IN, TR)
    % Includes biogas
    flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
    % 221   Emissions from Local Diesel use (DO, CO, IN, TR)
    % includes biodiesel
    flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
    % 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
    flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

    % 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
    %flowmat(:,223,2:4) = -1 * (0);
    % 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
    %flowmat(:,224,2:4) = -1 * (0);
    % 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
    %flowmat(:,225,2:4) = -1 * (0);
    % 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
    flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
    % 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
    flowmat(:,227,2:4) = -1 * (0);

    % 232   Total Emissions from Local Fuel Consumption
```

```
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);
```

```
% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%*********************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%*********************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%*************************************************************************
% Water
%*************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));
```

```
we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%************************************************************************
% Nitrogen
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
```

```
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%************************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%************************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass
```

```
I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%*********************************************************************
% Forestry Sector
%*********************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%*********************************************************************
% Food Sector
```

```
%*********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%*********************************************************************
% Energy Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
```

```
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%*********************************************************************
% Waste Management Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%*********************************************************************
% INDICATOR SCORES ARE CALCULATED                                    *
%*********************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
```

```
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%************************************************************************
%% NEW indicator
%************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;
```

```
%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
```

```
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;
```

```
      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%*************************************************************************
% Store data
%*************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
```

```
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%************************************************************************
% Estimations associated with land uses
%************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%************************************************************************
% ESTIMATION OF AREA for each designated land use
%************************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;
```

```
% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a
```

```
% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ********************************************************************
```

```
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%************************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%************************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% ************************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ************************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% ************************************************************************
% CALCULATION OF food consumption (by humans)
```

```
% ************************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
```

```
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
```

```
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
```

```
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%*********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
```

```
%*********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
```

```
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%*********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%*********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
```

```
        end
    end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    end
    % Correction for air cooling technology for natural gas power generation
    flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));
```

```
% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%*********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%*********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
```

```
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);
```

```
%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                    % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);              % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%*********************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%*********************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);
```

```
% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);
```

```
        % Limit evaporation to the amount of precipitation for
        % [forest grass crops] in mm/year
        for luse_et = 2:4
           if Q_Et(luse_et,4) > precip
           if Q_Et(luse_et,4) > precip
           if Q_Et(luse_et,4) > precip
           end
        end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;
```

```
% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
```

```
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    end
    % (b) sewer infiltration index
    sinfil = parause(55) * sinfil;

    % Runoff from pervious areas
    flowmat(:,77,:) = epa - ginfil;

    % Calculation of the fertilization mix (area-wise)
    fertmix

    tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
    tpa = statout(:,27); % total pervious area km2
    %tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
    Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
    % Forested areas accounts also for urban forests
    %*********************************************************************
    % FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
    %*********************************************************************
    % Creates a matrix of the proportion of non-fertilized area for each land
    % use with the size of the alternatives of fertilization
    fractnF = zeros(tfe,size(Areas,2));
    % [grass forestry crops]
    for a = 1:tfe;
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
    end

    % Specification of the fertilizer mix (in terms of area proportion) for
    % each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
    fe_mix = a_mix.* (ones(tfe,3) - fractnF);
    % fe_mix modified. non-fertilized area fraction is added.
    % fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
    fe_mix = cat(1,fractnF(1,:),fe_mix);
    % depth = [number of years]
    fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
    for a = 1:x % each year
        for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

    %************************************************************************
    % GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
```

```
%************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%************************************************************************
```

```
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%*********************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%*********************************************************************
% WATER OUTPUT CALCULATIONS
%*********************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%*********************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%*********************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
```

```
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
```

```
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%**********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%**********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
```

```
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%*********************************************************************
% Algae growing from WWTP effluent
%*********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
```

```
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end
```

```
%************************************************************************
% Emissions to air
%************************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%************************************************************************
% Activated sludge reactor
%************************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
```

```
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%*********************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%*********************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ====================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%=================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
```

```
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%************************************************************************
% BIOGAS FLOW
%************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%************************************************************************
% TREATED SLUDGE FLOW
%************************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
```

```
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*********************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
           parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
           parause(227) * log(infBOD) + ...
           parause(228) * log(effBOD) + ...
           parause(236) * log(100 * (1 - parause(64))) + ...
           parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*********************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
       parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
       parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
       parause(238) * log(statin(57)) + ...
       parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
```

```
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
```

```
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ************************************************************
% Calculation of live tree removals
% ************************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%*********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%*********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%*********************************************************************
```

```
% CALCULATION OF CARBON POOLS
%*********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
```

```
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%**********************************************************************
% CALCULATION of SAWMILLS input and output
%**********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
       if a == 1
           statout(a,34) = smcap1(a) + smcap0;
       end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%**********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%**********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;
```

```
%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
```

```
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
%*********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%*********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%*********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
```

```
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%*******************************************************************
```

```
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%*********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
```

```
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%*****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%*****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
```

```
        for b = 1:3 % [N P C]
           rec_fert(a,b) = parause2(bB + b)./10;
           rec_fert(a,b) = parause2(bB + b)./10;
           rec_fert(a,b) = parause2(bB + b)./10;
        end
        bB = bB + b + 1;
    end
    %*****************************************************************
    % Nutrient Content for each type of fertilizer IF SS PL CM SM CP
    %*****************************************************************
    % First we calculate the water content (moisture) of manure after storage
    % & handling (estimated in manure.m) as a fraction w/w of total mass,
    % nutrient content as w/w fraction of DM, and energy in kWh/tonne
    fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

    % Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
    % Column of zeros for IF
    for a = 1:nmat
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    end
    % Now data is corrected to reflect fraction of mass. For compost we use the
    % typical compost composition entered by user
    % Dry matter in tonne/y
    fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
    % We convert first column from Water content Mgal/d to w/w of total mass
    fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

    % Nutrient content corrected to w/w of dry matter
    for a = 2:4
        fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
        fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
        fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    end
    % Energy content corrected to kWh per tonne (fresh matter)
    fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

    % fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
    % fraction as from 1 to 5 are.
    % Data is added for IF (not calculated above) from user input
    % Assumed moisture 0.5%
    fm_cont(isnan(fm_cont)) = 0;
    fm_cont(:,1,1) = ones(x,1) * 0.005;
    fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

    % Data of ideal compost is added from user input
    fm_cont(:,6,1) = parause2(128)/100;
    fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
    % energy calculated from ultimate analysis
```

```
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
       + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
       - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
```

```
    end
    % Calculates the flow of all constituents in the fertilizer applied
    % by type of fertilizer as calculated by the limiting nutrient.
    % fm_cont = years x [IF SS PL CM SM CP] x species
    for a = 1:x % year
        for b = 0:3 % [IF SS PL CM]
            % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
            flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
            flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
            flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
            flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
            flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
            flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
            % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
            flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
            flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
            flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
            flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
            flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
            flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
            % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
            flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
            flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
            flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
            flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
            flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
            flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
            % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
            flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
            flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
            flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
            flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
            flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
            flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        end
    end
    % Swine Manure (SM) applied ...added last minute. flowmat(:,183
    for a = 1:x % year
        flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
        flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
        flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
        flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
        flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
        flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
    end
    % Compost (CP) applied ...added last minute. flowmat(:,316
    for a = 1:x % year
        flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
        flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
        flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
        flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
        flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
        flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
    end
    % verification of nutrient application and calculation of the required
    % extra nutrient (non-limiting) that will be supplied as inorganic
    % fertilizer, only if in deficit.
    if modset(7) == 1;
        nlimN = 2; %Phosphorus is the non-limiting Nutrient
    end
    % nutrient fertilizer actually applied, tonne/y N or P
```

```
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%************************************************************************
%
```

```
% APPLICATION PER LAND USE. Added August 20010
%
%
%*********************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
```

```
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%***************************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%***************************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%************************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
```

```
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%************************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%************************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%************************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
```

```
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
```

```
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%*********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
```

```
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
```

```
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%**************************************************************************
% CONVERSIONS
%**************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
```

```
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%*********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
```

```
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
```

```
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
```

```
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  *************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  *************************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
end
```

```
% Total meat produced
flowmat(:,160,:) = sum(P_meat,2);

% Total animal (meat) mass accumulated due to increase or decrease of
% livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
L_acc = zeros(x,3,nmat);
% Average livestock growth in tonne/y, based on growth of thousand
% heads: 'liveg' calculated in 'livepop.m'
for b = 0:2
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
end
% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
end
% Total livestock accumulation
flowmat(:,306,:) = sum(L_acc,2);

% DAIRY  **************************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
```

```
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);            % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
```

```
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);           % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ************************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  *************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
```

```
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
```

```
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*********************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*********************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));
```

```
%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;
```

```
%************************************************************************
% BIOGAS FLOW
%************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%************************************************************************
% FOOD SLUDGE FLOW
%************************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%************************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%************************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
```

```
        flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
    % Emission from food products in MSW incineration
        flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
    % Emission from sewage sludge in incineration
        flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
    % Emission from wood products in MSW incineration
        flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
    % Emission from food products in MSW incineration
        flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
    % Emission from sewage sludge in incineration
        flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
    % Emission from wood products in MSW incineration
        flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
    % Emission from food products in MSW incineration
        flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
    % Emission from sewage sludge in incineration
        flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
    % Emission from wood products in MSW incineration
        flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
    % Emission from food products in MSW incineration
        flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
    % Emission from sewage sludge in incineration
        flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
    % Emission from wood products in MSW incineration
        flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
    % Emission from food products in MSW incineration
        flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
    % Emission from sewage sludge in incineration
        flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
    % Emission from wood products in MSW incineration
        flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
    % Emission from food products in MSW incineration
        flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
    % Emission from sewage sludge in incineration
        flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
    % Emission from wood products in MSW incineration
        flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
    % Emission from food products in MSW incineration
        flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
    % Emission from sewage sludge in incineration
        flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
    end
    % Energy content of incineration gases, complete
    flowmat(:,143,5) = 0;
    flowmat(:,280,5) = 0;
    flowmat(:,336,5) = 0;

    %***********************************************************************
    % RECOVERY OF ENERGY FROM MSW COMBUSTION
    %***********************************************************************
    % Energy generated is calculated based on a the energy content of materials
    % and a combustion efficiency
    % Energy from MSW (wood + food + sludge)
    % Based on efficiency
    flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
    % Only sludge
    flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

    %***********************************************************************
    % CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
    %***********************************************************************
```

```
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%*********************************************************************
% First part of the energy module. second part is 'energys2.m'
%*********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
```

```
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%************************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%************************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
```

```
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%*********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
```

```
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%*********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%*********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
```

```
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                           % content in raw coal (dry basis)
                           % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%************************************************************************
% Algae growing from coal plant emissions
%************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
```

```
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%**************************************************************************
% Production of biodiesel
%**************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;
```

```
% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%**************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%**************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
```

```
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%**********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%**********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**********************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%**********************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%**********************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%**********************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro
```

```
py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
```

```
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
```

```
        pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
    % Ultimate composition on a dry basis (percentage)
        pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
            pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
    % HHV in MJ/kg of dry mass
        pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
            + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
            - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
    % Energy in kWh/y
        pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
    % Dry mass based on ultimate composition
        pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
    % Ultimate composition on a dry basis (percentage)
        pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
            pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
    % HHV in MJ/kg of dry mass
        pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
            + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
            - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
    % Energy in kWh/y
        pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
    % Dry mass based on ultimate composition
        pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
    % Ultimate composition on a dry basis (percentage)
        pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
            pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
    % HHV in MJ/kg of dry mass
        pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
            + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
            - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
    % Energy in kWh/y
        pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
    % Dry mass based on ultimate composition
        pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
    % Ultimate composition on a dry basis (percentage)
        pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
            pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
    % HHV in MJ/kg of dry mass
        pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
            + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
            - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
    % Energy in kWh/y
        pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
    end
    %Check energy by mass partition
    % [condensate solids gas]
    %El = pyenergy(:,2) * pyYield(1,10);
    %Ec = pyenergy(:,3) * pyYield(2,10);
    %Eg = pyenergy(:,4) * pyYield(3,10);
    %El + Ec + Eg;
    %alternative for calculating mass of each phase (using mass yield factors)
    % [condensate solids gas]
    pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
    pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
    pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

    % Checking: sum(pyFlows(:,1:8,1),2)
    % sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
    % pyFlows(:,10,1) + py_feed(:,:,7)
    % Pyrolysis outputs (in flowmat format)
```

```
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                     % Liquid phase
flowmat(:,267,7)    = 0;                     % Char phase
flowmat(:,266,7)    = 0;                     % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                    % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                    % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                    % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%**********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%**********************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%**********************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%**********************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%**********************************************************************
% LANDFILLING
%**********************************************************************
% Estimation of materials landfilled MSW
```

```
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
```

```
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%*********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%*********************************************************************
% Gas generated from:
```

```
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%*********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%*********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
```

```
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    end
    % Resulting composted fertilizer - CORRECTING TOTAL MASS
    % Substracting water and nutrient loss
    flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
    % check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
    % Energy content calculated based on Channiwala 2002
    % HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
    % HHV in MJ/kg of dry mass
    % Estimation for H S O Ash is based on their relationship to carbon content
    % in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
    % and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
    % compared to fresh manure ~20-30%. So we double of Singh's values,
    % Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
    % with the ~51% reduction in Carbon content during composting that promotes
    % a larger Ash/C ratio.
    % Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
    % check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
    UAn     = zeros(x,6);    % percentage [C H S O N Ash]
    Drym    = flowmat(:,315,6) - flowmat(:,315,7);
    UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
    UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
    UAn(:,5) = flowmat(:,315,2)./Drym * 100;
    UAn(:,6) = UAn(:,1) * 1.4;
    UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

    %Energy in MJ/kg
    comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
            + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
            - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

    % Energy in kWh/y
    flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

    % Exported or imported organic fertilizer. Includes composted material and
    % solids from food waste anaerobic digestion
```

```
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%****************************************************************
% EXPORTS/IMPORTS
%****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
```

```
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%**********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%**********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
```

```
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%*********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%*********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
```

```
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    end
    % Biomass emission factor tonne/kwh (assumed as wood)
    bitkwh = parause2(129:131) * parause2(132)^-1;
    bitkwh(2) = 0; % Phosphorus corrected to be zero

    %%% Correction. Natural gas, gasoline, and diesel are assumed to
    %%% undergo 100% oxidation, i.e. all C and N is released as emission
    % Natural gas emission factor tonne/kwh (100% oxidation)
    %ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
    % Gasoline emission factor tonne/kwh (100% oxidation)
    %gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
    % Diesel emission factor tonne/kwh (100% oxidation)
    %ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
    % 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
    % Coal combustion generates CCP and emissions, partitioning input materials
    tol_coal_em = zeros(x,3);
    for a = 1:x
        tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
    end
    flowmat(:,219,2:4) = -1 * (tol_coal_em);

    % 220   Emissions from Local NG use (DO, CO, IN, TR)
    % Includes biogas
    flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
    % 221   Emissions from Local Diesel use (DO, CO, IN, TR)
    % includes biodiesel
```

```
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*****************************************************************
```

```
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%*********************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%*********************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));
```

```
% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%************************************************************************
% Water
%************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%************************************************************************
% Nitrogen
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
```

```
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%-------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%-------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero
```

```
%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%************************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%************************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%************************************************************************
% Forestry Sector
%************************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);
```

```
flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%*********************************************************************
% Food Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%*********************************************************************
% Energy Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;
```

```
Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power geneneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%**********************************************************************
% Waste Management Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%**********************************************************************
% INDICATOR SCORES ARE CALCULATED                                     *
%**********************************************************************
```

```
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%************************************************************************
%% NEW indicator
%************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
```

```
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
```

```
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
```

```
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end
```

```
    if modset(12) == 3
    end

topo = topo';
%
%*************************************************************************
% Store data
%*************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%*************************************************************************
% Estimations associated with land uses
%*************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%*************************************************************************
% ESTIMATION OF AREA for each designated land use
%*************************************************************************
```

```
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
```

```
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
```

```
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%************************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%************************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);
```

```
% ************************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ************************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
       flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
           - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% ************************************************************************
% CALCULATION OF food consumption (by humans)
% ************************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
```

```
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
```

```
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
```

```
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
```

```
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%*********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%*********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
```

```
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%*********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%*********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
```

```
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
```

```
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
end
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%**********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%**********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
```

```
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);
```

```
flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                   % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);             % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%*********************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%*********************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;
```

```
% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
```

```
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];
```

```
% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
```

```
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
end
% (b) sewer infiltration index
sinfil = parause(55) * sinfil;

% Runoff from pervious areas
flowmat(:,77,:) = epa - ginfil;

% Calculation of the fertilization mix (area-wise)
fertmix

tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
tpa = statout(:,27); % total pervious area km2
%tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
% Forested areas accounts also for urban forests
%*******************************************************************
% FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
%*******************************************************************
% Creates a matrix of the proportion of non-fertilized area for each land
% use with the size of the alternatives of fertilization
fractnF = zeros(tfe,size(Areas,2));
% [grass forestry crops]
for a = 1:tfe;
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
end

% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = a_mix.* (ones(tfe,3) - fractnF);
% fe_mix modified. non-fertilized area fraction is added.
% fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = cat(1,fractnF(1,:),fe_mix);
% depth = [number of years]
```

```
fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
for a = 1:x % each year
    for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%************************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
```

```
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%*********************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%*********************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                    % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%*********************************************************************
% WATER OUTPUT CALCULATIONS
%*********************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%*********************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%*********************************************************************
% Evapotranspiration from PA + IA
```

```
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
```

```
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%*********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%*********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
```

```
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
```

```
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%**********************************************************************
% Algae growing from WWTP effluent
%**********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
```

```
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%*********************************************************************
% Emissions to air
%*********************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%*********************************************************************
% Activated sludge reactor
```

```
%*********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%*********************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%*********************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ====================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
```

```
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%**********************************************************************
% BIOGAS FLOW
%**********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%**********************************************************************
% TREATED SLUDGE FLOW
%**********************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
```

```
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*********************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);
```

```
% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*****************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
```

```
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ***********************************************************
% Calculation of live tree removals
% ***********************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*****************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*****************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber
```

```
noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%***********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%***********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%***********************************************************************
% CALCULATION OF CARBON POOLS
%***********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;
```

```
% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
        if a == 1
            statout(a,34) = smcap1(a) + smcap0;
        end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
```

```
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
```

```
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
%*********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;
```

```
%**********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%**********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
```

```
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%*********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%*********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
```

```
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%*****************************************************************
% Recommended Nutrient FOR EACH LAND USE
```

```
%*****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
end
%*****************************************************************
% Nutrient Content for each type of fertilizer IF SS PL CM SM CP
%*****************************************************************
% First we calculate the water content (moisture) of manure after storage
% & handling (estimated in manure.m) as a fraction w/w of total mass,
% nutrient content as w/w fraction of DM, and energy in kWh/tonne
fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

% Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
% Column of zeros for IF
for a = 1:nmat
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
end
% Now data is corrected to reflect fraction of mass. For compost we use the
% typical compost composition entered by user
% Dry matter in tonne/y
fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
% We convert first column from Water content Mgal/d to w/w of total mass
fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

% Nutrient content corrected to w/w of dry matter
```

```
for a = 2:4
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
end
% Energy content corrected to kWh per tonne (fresh matter)
fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

% fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
% fraction as from 1 to 5 are.
% Data is added for IF (not calculated above) from user input
% Assumed moisture 0.5%
fm_cont(isnan(fm_cont)) = 0;
fm_cont(:,1,1) = ones(x,1) * 0.005;
fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
```

```
        if modset(7) == 1
            FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
        end
        req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
    end
    % Preventing NaN
    AreInf = (isinf(req_fert)==1);
    req_fert(AreInf) = 0;

    % Actual amount of fertilizer applied for each land use and type of
    % fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
    % Pre-allocating matrix space
    fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
    fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
    fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

    % Actual fertilizer mass applied as tonne/y of dry matter
    for a = 1:x % each year
        fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
    end
    % Calculates the flow of all constituents in the fertilizer applied
    % by type of fertilizer as calculated by the limiting nutrient.
    % fm_cont = years x [IF SS PL CM SM CP] x species
    for a = 1:x % year
        for b = 0:3 % [IF SS PL CM]
            % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
            flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
            flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
            flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
            flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
            flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
            flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
            % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
            flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
            flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
            flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
            flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
            flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
            flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
            % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
            flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
            flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
            flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
            flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
            flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
            flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
            % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
            flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
            flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
            flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
            flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
            flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
            flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        end
    end
    % Swine Manure (SM) applied ...added last minute. flowmat(:,183
    for a = 1:x % year
        flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
        flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
        flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
```

```
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%*************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
```

```
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%**************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%**************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
```

```
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%******************************************************************
% MANURE distribution
%******************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%**********************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%**********************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);
```

```
% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%*********************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%*********************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%*********************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%*********************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%*********************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%*********************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year
```

```
% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
```

```
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%*********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
```

```
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%*********************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%*********************************************************************
```

```
 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%************************************************************************
% CONVERSIONS
%************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner
```

```
En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%**************************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%**************************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);
```

```
% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
```

```
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
```

```
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  ************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  ************************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
```

```
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    end
    % Total meat produced
    flowmat(:,160,:) = sum(P_meat,2);

    % Total animal (meat) mass accumulated due to increase or decrease of
    % livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
    L_acc = zeros(x,3,nmat);
    % Average livestock growth in tonne/y, based on growth of thousand
    % heads: 'liveg' calculated in 'livepop.m'
    for b = 0:2
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    end
    % [year] x [Cattle Poultry Pigs] x [specie]
    for b = 0:4:8
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    end
    % Total livestock accumulation
    flowmat(:,306,:) = sum(L_acc,2);

    % DAIRY  **************************************************************
```

```
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ********************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
```

```
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);           % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);           % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ***********************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
```

```
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;    % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);            % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  ************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
```

```
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*********************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*********************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
```

```
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
```

```
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%**************************************************************************
% BIOGAS FLOW
%**************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%**************************************************************************
% FOOD SLUDGE FLOW
%**************************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
```

```
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%**********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%**********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
```

```
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%*********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%*********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%*********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%*********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%*********************************************************************
% First part of the energy module. second part is 'energys2.m'
%*********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
```

```
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%**************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%**************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
```

```
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%*********************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%*********************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%*********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
```

```
%************************************************************************
% Estimation of flows for power generation
%************************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%************************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%************************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
```

```
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%**********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%**********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                           % content in raw coal (dry basis)
                           % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));
```

```
%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%************************************************************************
% Algae growing from coal plant emissions
%************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%************************************************************************
% Production of biodiesel
%************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
```

```
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%**************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%**************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);
```

```
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%**********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%**********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**********************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%**********************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
```

```
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%************************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%************************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
```

```
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
```

```
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
```

```
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

% Checking: sum(pyFlows(:,1:8,1),2)
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                    % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                    % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                    % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
```

```
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%***********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%***********************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%***********************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%***********************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%***********************************************************************
% LANDFILLING
%***********************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
```

```
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));
```

```
% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%***********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%***********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%***********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%***********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
```

```
        end
    end
    for a = 1:nmat
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    end
    % Resulting composted fertilizer - CORRECTING TOTAL MASS
    % Substracting water and nutrient loss
    flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
    % check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
    % Energy content calculated based on Channiwala 2002
    % HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
    % HHV in MJ/kg of dry mass
    % Estimation for H S O Ash is based on their relationship to carbon content
    % in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
    % and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
    % compared to fresh manure ~20-30%. So we double of Singh's values,
    % Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
    % with the ~51% reduction in Carbon content during composting that promotes
    % a larger Ash/C ratio.
```

```
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
UAn     = zeros(x,6);    % percentage [C H S O N Ash]
Drym    = flowmat(:,315,6) - flowmat(:,315,7);
UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = flowmat(:,315,2)./Drym * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
       + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
       - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%*****************************************************************
% EXPORTS/IMPORTS
%*****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%*****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%*****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*****************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
```

```
%**********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%**********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%**********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
```

```
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
```

```
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%*********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%*********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
```

```
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));
```

```
% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
```

```
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%************************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%************************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%************************************************************************
% Water
%************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%************************************************************************
% Nitrogen
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
```

```
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);
```

```
% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%-------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%-------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%*************************************************************************
% Rest - no healthy emissions defined
%*************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%*************************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%*************************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%*************************************************************************
```

```
% Forestry Sector
%*********************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%*********************************************************************
% Food Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
```

```
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%**********************************************************************
% Energy Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%**********************************************************************
% Waste Management Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
```

```
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%************************************************************************
% INDICATOR SCORES ARE CALCULATED                                       *
%************************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%************************************************************************
%% NEW indicator
%************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
```

```
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
```

```
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
```

```
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo
```

```
p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%*************************************************************************
% Store data
%*************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%*************************************************************************
% Estimations associated with land uses
%*************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
```

```
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%*********************************************************************
% ESTIMATION OF AREA for each designated land use
%*********************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
```

```
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
```

```
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%************************************************************************
```

```
% SEWAGE AND MUNICIPAL SOLID WASTE
%*********************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% *********************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% *********************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% *********************************************************************
% CALCULATION OF food consumption (by humans)
% *********************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%*********************************************************************
% FOOD CONSUMPTION
%*********************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
```

```
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
```

```
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
```

```
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
```

```
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%**************************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%**************************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
```

```
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%*********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
```

```
%*********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
```

```
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
end
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%*********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%*********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
```

```
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);
```

```
% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                   % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
```

```
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);             % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%************************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%************************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
```

```
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
```

```
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
```

```
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
end
% (b) sewer infiltration index
sinfil = parause(55) * sinfil;

% Runoff from pervious areas
flowmat(:,77,:) = epa - ginfil;

% Calculation of the fertilization mix (area-wise)
fertmix

tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
tpa = statout(:,27); % total pervious area km2
%tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
% Forested areas accounts also for urban forests
%*****************************************************************
% FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
%*****************************************************************
% Creates a matrix of the proportion of non-fertilized area for each land
% use with the size of the alternatives of fertilization
fractnF = zeros(tfe,size(Areas,2));
% [grass forestry crops]
```

```
for a = 1:tfe;
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
end

% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = a_mix.* (ones(tfe,3) - fractnF);
% fe_mix modified. non-fertilized area fraction is added.
% fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = cat(1,fractnF(1,:),fe_mix);
% depth = [number of years]
fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
for a = 1:x % each year
    for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%*************************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%*************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
```

```
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%**************************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%**************************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%**************************************************************************
% WATER OUTPUT CALCULATIONS
%**************************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
```

```
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%**************************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%**************************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
```

```
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%*********************************************************************
```

```
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%*********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C **********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
```

```
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%**********************************************************************
% Algae growing from WWTP effluent
%**********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
```

```
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%**********************************************************************
% Emissions to air
%**********************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));
```

```
biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%*********************************************************************
% Activated sludge reactor
%*********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%*********************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%*********************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
```

```
% ================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%**********************************************************************
% BIOGAS FLOW
%**********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%**********************************************************************
% TREATED SLUDGE FLOW
%**********************************************************************
```

```
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*******************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
```

```
ene_wwtp =  parause(225) + ...
           parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
           parause(227) * log(infBOD) + ...
           parause(228) * log(effBOD) + ...
           parause(236) * log(100 * (1 - parause(64))) + ...
           parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*****************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
```

```
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ********************************************************
% Calculation of live tree removals
% ********************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
```

```
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%**********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%**********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%**********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%**********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%**********************************************************************
% CALCULATION OF CARBON POOLS
%**********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
```

```
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;
```

```
%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
       if a == 1
           statout(a,34) = smcap1(a) + smcap0;
       end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;
```

```
% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
```

```
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
%*********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
```

```
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%*********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%*********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
```

```
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
```

```
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%*****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%*****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
end
%*****************************************************************
% Nutrient Content for each type of fertilizer IF SS PL CM SM CP
%*****************************************************************
% First we calculate the water content (moisture) of manure after storage
% & handling (estimated in manure.m) as a fraction w/w of total mass,
% nutrient content as w/w fraction of DM, and energy in kWh/tonne
fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

% Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
% Column of zeros for IF
for a = 1:nmat
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
```

```
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
end
% Now data is corrected to reflect fraction of mass. For compost we use the
% typical compost composition entered by user
% Dry matter in tonne/y
fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
% We convert first column from Water content Mgal/d to w/w of total mass
fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

% Nutrient content corrected to w/w of dry matter
for a = 2:4
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
end
% Energy content corrected to kWh per tonne (fresh matter)
fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

% fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
% fraction as from 1 to 5 are.
% Data is added for IF (not calculated above) from user input
% Assumed moisture 0.5%
fm_cont(isnan(fm_cont)) = 0;
fm_cont(:,1,1) = ones(x,1) * 0.005;
fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
       + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
       - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%****************************************************************
% NUTRIENT APPLICATION RATE
%****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
```

```
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
```

```
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%*************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
```

```
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%**************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%**************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
```

```
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
```

```
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%************************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%************************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%************************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%************************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%************************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%************************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
```

```
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
```

```
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%**********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
```

```
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
```

```
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%************************************************************************
%****************** (originally calculated in waters) *******************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%************************************************************************
% CONVERSIONS
%************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
```

```
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%**********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
```

```
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
```

```
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
```

```
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  ************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
```

```
                              % Chattahoochee Watershed
% MEAT  **********************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
end
% Total meat produced
flowmat(:,160,:) = sum(P_meat,2);

% Total animal (meat) mass accumulated due to increase or decrease of
% livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
L_acc = zeros(x,3,nmat);
% Average livestock growth in tonne/y, based on growth of thousand
% heads: 'liveg' calculated in 'livepop.m'
for b = 0:2
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
end
% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
```

```
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
end
% Total livestock accumulation
flowmat(:,306,:) = sum(L_acc,2);

% DAIRY  ************************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
```

```
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);            % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);            % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ***********************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
```

```
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);            % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  ************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
```

```
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% -------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%--------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*************************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*************************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
```

```
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%**********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%**********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%**********************************************************************
% Waste reducing methods
%**********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);
```

```
% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%*********************************************************************
% FOOD SLUDGE FLOW
%*********************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;
```

```
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%*********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%*********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
```

```
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%**********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%**********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%**********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%**********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%**********************************************************************
% First part of the energy module. second part is 'energys2.m'
%**********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR
```

```
% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%*********************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
```

```
%************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%************************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%************************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%************************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%************************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;
```

```
% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%*********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%*********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
```

```
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                            % content in raw coal (dry basis)
                            % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
```

```
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%************************************************************************
% Algae growing from coal plant emissions
%************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%************************************************************************
% Production of biodiesel
%************************************************************************
```

```
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%**************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%**************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
```

```
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%*********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%*********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
```

```
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**************************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%**************************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%**************************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%**************************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];
```

```
% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%**********************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%**********************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';
```

```
% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
```

```
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

% Checking: sum(pyFlows(:,1:8,1),2)
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;        % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;        % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;        % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
```

```
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                    % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                    % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                    % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%**********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%**********************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%**********************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%**********************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%**********************************************************************
% LANDFILLING
%**********************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
```

```
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
```

```
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%*********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%*********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%*********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%*********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
```

```
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
```

```
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
end
% Resulting composted fertilizer - CORRECTING TOTAL MASS
% Substracting water and nutrient loss
flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
% check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
% and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
% compared to fresh manure ~20-30%. So we double of Singh's values,
% Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
% with the ~51% reduction in Carbon content during composting that promotes
% a larger Ash/C ratio.
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
UAn     = zeros(x,6);    % percentage [C H S O N Ash]
Drym    = flowmat(:,315,6) - flowmat(:,315,7);
UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = flowmat(:,315,2)./Drym * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%****************************************************************
% EXPORTS/IMPORTS
%****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
```

```
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%**********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%**********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
```

```
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%*********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
```

```
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%**********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%**********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
```

```
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
```

```
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
```

```
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%*********************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%*********************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%*********************************************************************
% Water
%*********************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%*********************************************************************
% Nitrogen
%*********************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
```

```
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
```

```
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%*********************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%*********************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
```

```
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%************************************************************************
% Forestry Sector
%************************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%************************************************************************
% Food Sector
%************************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;
```

```
ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%**************************************************************************
% Energy Sector
%**************************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
```

```
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%*********************************************************************
% Waste Management Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%*********************************************************************
% INDICATOR SCORES ARE CALCULATED                                    *
%*********************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
```

```
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%************************************************************************
%% NEW indicator
%************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
```

```
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
```

```
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
```

```
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%************************************************************************
% Store data
%************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%************************************************************************
% Estimations associated with land uses
%************************************************************************
landuse
```

```
land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%*********************************************************************
% ESTIMATION OF AREA for each designated land use
%*********************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);
```

```
% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
```

```
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
```

```
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%**********************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%**********************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% **********************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% **********************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% **********************************************************************
% CALCULATION OF food consumption (by humans)
% **********************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
```

```
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
```

```
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';
```

```
% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
```

```
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%*********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%*********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
```

```
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
```

```
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%*********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%*********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
```

```
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
end
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
```

```
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%*********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%*********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
```

```
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%**********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%**********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%**********************************************************************
% WATER INPUTS ESTIMATION
%**********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                    % mm/year
```

```
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                    % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);              % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%************************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%************************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);
```

```
% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            end
```

```
        end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
```

```
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
end
% (b) sewer infiltration index
sinfil = parause(55) * sinfil;

% Runoff from pervious areas
```

```
flowmat(:,77,:) = epa - ginfil;

% Calculation of the fertilization mix (area-wise)
fertmix

tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
tpa = statout(:,27); % total pervious area km2
%tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
% Forested areas accounts also for urban forests
%*********************************************************************
% FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
%*********************************************************************
% Creates a matrix of the proportion of non-fertilized area for each land
% use with the size of the alternatives of fertilization
fractnF = zeros(tfe,size(Areas,2));
% [grass forestry crops]
for a = 1:tfe;
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
end

% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = a_mix.* (ones(tfe,3) - fractnF);
% fe_mix modified. non-fertilized area fraction is added.
% fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = cat(1,fractnF(1,:),fe_mix);
% depth = [number of years]
fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
for a = 1:x % each year
    for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%************************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters
```

```
% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%*********************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%*********************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
```

```
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%************************************************************************
% WATER OUTPUT CALCULATIONS
%************************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%************************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%************************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
```

```
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a
```

```
% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%*********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%*********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
```

```
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%*********************************************************************
% Algae growing from WWTP effluent
%*********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
```

```
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%*********************************************************************
% Emissions to air
%*********************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
```

```
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%*********************************************************************
% Activated sludge reactor
%*********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
```

```
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%**********************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%**********************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ====================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
```

```
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%************************************************************************
% BIOGAS FLOW
%************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%************************************************************************
% TREATED SLUDGE FLOW
%************************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
```

```
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*******************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*******************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
```

```
%************************************************************************
% Consistency verification
%************************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ********************************************************
% Calculation of live tree removals
% ********************************************************
```

```
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%*********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%*********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%*********************************************************************
% CALCULATION OF CARBON POOLS
%*********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y
```

```
% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
```

```
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
        if a == 1
            statout(a,34) = smcap1(a) + smcap0;
        end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
```

```
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);
```

```
% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%**********************************************************************
% ATMOSPHERIC PROCESSES
%**********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%*********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%*********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
```

```
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%*********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%*********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
```

```
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
```

```
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%*****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%*****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
end
%*****************************************************************
```

```
% Nutrient Content for each type of fertilizer IF SS PL CM SM CP
%*******************************************************************
% First we calculate the water content (moisture) of manure after storage
% & handling (estimated in manure.m) as a fraction w/w of total mass,
% nutrient content as w/w fraction of DM, and energy in kWh/tonne
fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

% Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
% Column of zeros for IF
for a = 1:nmat
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
end
% Now data is corrected to reflect fraction of mass. For compost we use the
% typical compost composition entered by user
% Dry matter in tonne/y
fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
% We convert first column from Water content Mgal/d to w/w of total mass
fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

% Nutrient content corrected to w/w of dry matter
for a = 2:4
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
end
% Energy content corrected to kWh per tonne (fresh matter)
fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

% fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
% fraction as from 1 to 5 are.
% Data is added for IF (not calculated above) from user input
% Assumed moisture 0.5%
fm_cont(isnan(fm_cont)) = 0;
fm_cont(:,1,1) = ones(x,1) * 0.005;
fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
```

```
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%****************************************************************
% NUTRIENT APPLICATION RATE
%****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
```

```
            flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
            flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
            flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
            flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
            flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
            % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
            flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
            flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
            flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
            flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
            flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
            flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
            % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
            flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
            flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
            flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
            flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
            flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
            flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
            % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
            flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
            flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
            flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
            flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
            flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
            flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        end
    end
    % Swine Manure (SM) applied ...added last minute. flowmat(:,183
    for a = 1:x % year
        flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
        flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
        flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
        flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
        flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
        flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
    end
    % Compost (CP) applied ...added last minute. flowmat(:,316
    for a = 1:x % year
        flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
        flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
        flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
        flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
        flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
        flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
    end
    % verification of nutrient application and calculation of the required
    % extra nutrient (non-limiting) that will be supplied as inorganic
    % fertilizer, only if in deficit.
    if modset(7) == 1;
        nlimN = 2; %Phosphorus is the non-limiting Nutrient
    end
    % nutrient fertilizer actually applied, tonne/y N or P
    Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
    Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
    for a = 1:x % year
        for b = 1:3 % [grass forest crops]
           Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
           Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
           Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
           Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
```

```
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
```

```
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
```

```
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%************************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%************************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%*********************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%*********************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%*********************************************************************
```

```
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%**********************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%**********************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%**********************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;
```

```
% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);
```

```
% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%**********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
```

```
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
```

```
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%**************************************************************************
% CONVERSIONS
%**************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
```

```
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%*********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
```

```
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
```

```
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
```

```
        flowmat(:,175,:) + pet_food;

    % check that livestock is not excreting more than what is eating
    % f=0    f=1    f=2 [cattle poultry swine]
    % excreting/feed
    % flowmat(:,309+f,2)./flowmat(:,173+f,2)
    % flowmat(:,309+f,3)./flowmat(:,173+f,3)
    if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
        end
    else
        emat(19) = 0;
    end
    %***********************************************************************
    % PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
    %***********************************************************************
    % FISH  *************************************************************
    flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                               % Chattahoochee Watershed
    % MEAT  *************************************************************
    % Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
    % from livestock inventory at the begining of the year
    P_meat = zeros(x,3,nmat);
    % [years] x [Cattle Poultry Pigs]
    for b = 0:2
        P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
        P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
        P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    end

    % [year] x [Cattle Poultry Pigs] x [specie]
    for b = 0:4:8
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    end
    % Total meat produced
    flowmat(:,160,:) = sum(P_meat,2);

    % Total animal (meat) mass accumulated due to increase or decrease of
    % livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
    L_acc = zeros(x,3,nmat);
    % Average livestock growth in tonne/y, based on growth of thousand
    % heads: 'liveg' calculated in 'livepop.m'
```

```
for b = 0:2
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
end
% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
end
% Total livestock accumulation
flowmat(:,306,:) = sum(L_acc,2);

% DAIRY  *************************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
```

```
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);            % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
```

```
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);            % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ************************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);            % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  *************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);
```

```
nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*********************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*********************************************************************
% ** FOOD **
% [fruits cereal veggies]
```

```
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
```

```
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
```

```
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%**********************************************************************
% FOOD SLUDGE FLOW
%**********************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%**********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%**********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
```

```
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%*********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%*********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%*********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%*********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%**********************************************************************
% First part of the energy module. second part is 'energys2.m'
%**********************************************************************
```

```
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
```

```
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%************************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%************************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
```

```
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%*********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y
```

```
%**********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%**********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%**********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%**********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
```

```
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                            % content in raw coal (dry basis)
                            % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%************************************************************************
% Algae growing from coal plant emissions
%************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);
```

```
    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%**************************************************************************
% Production of biodiesel
%**************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;
```

```
% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%*********************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%*********************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);
```

```
%*********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%*********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%*********************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%*********************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%*********************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%*********************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
```

```
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
```

```
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
```

```
    % Energy in kWh/y
      pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
    % Dry mass based on ultimate composition
      pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
    % Ultimate composition on a dry basis (percentage)
      pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
          pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
    % HHV in MJ/kg of dry mass
      pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
          + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
          - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
    % Energy in kWh/y
      pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
    % Dry mass based on ultimate composition
      pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
    % Ultimate composition on a dry basis (percentage)
      pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
          pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
    % HHV in MJ/kg of dry mass
      pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
          + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
          - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
    % Energy in kWh/y
      pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
    % Dry mass based on ultimate composition
      pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
    % Ultimate composition on a dry basis (percentage)
      pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
          pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
    % HHV in MJ/kg of dry mass
      pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
          + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
          - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
    % Energy in kWh/y
      pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
    end
    %Check energy by mass partition
    % [condensate solids gas]
    %El = pyenergy(:,2) * pyYield(1,10);
    %Ec = pyenergy(:,3) * pyYield(2,10);
    %Eg = pyenergy(:,4) * pyYield(3,10);
    %El + Ec + Eg;
    %alternative for calculating mass of each phase (using mass yield factors)
    % [condensate solids gas]
    pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
    pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
    pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

    % Checking: sum(pyFlows(:,1:8,1),2)
    % sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
    % pyFlows(:,10,1) + py_feed(:,:,7)
    % Pyrolysis outputs (in flowmat format)
    % Water content in mass units tonne/y (assumed to be lost during the
    % pyrolysis and upgrading process)
    flowmat(:,265,7)    = 0;                    % Liquid phase
    flowmat(:,267,7)    = 0;                    % Char phase
    flowmat(:,266,7)    = 0;                    % gas phase
    % Water content in volume units Mgal/d
    flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
    flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
```

```
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                  % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                  % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                  % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                  % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                  % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                  % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                  % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                  % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                  % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                     % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                     % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                     % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%*********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%*********************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%*********************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%*********************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%*********************************************************************
% LANDFILLING
%*********************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
```

```
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
```

```
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%*********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%*********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%*********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%*********************************************************************
```

```
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
```

```
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    end
    % Resulting composted fertilizer - CORRECTING TOTAL MASS
    % Substracting water and nutrient loss
    flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
    % check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
    % Energy content calculated based on Channiwala 2002
    % HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
    % HHV in MJ/kg of dry mass
    % Estimation for H S O Ash is based on their relationship to carbon content
    % in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
    % and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
    % compared to fresh manure ~20-30%. So we double of Singh's values,
    % Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
    % with the ~51% reduction in Carbon content during composting that promotes
    % a larger Ash/C ratio.
    % Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
    % check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
    UAn     = zeros(x,6);    % percentage [C H S O N Ash]
    Drym    = flowmat(:,315,6) - flowmat(:,315,7);
    UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
    UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
    UAn(:,5) = flowmat(:,315,2)./Drym * 100;
    UAn(:,6) = UAn(:,1) * 1.4;
    UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

    %Energy in MJ/kg
    comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
            + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
            - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

    % Energy in kWh/y
    flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

    % Exported or imported organic fertilizer. Includes composted material and
    % solids from food waste anaerobic digestion
    % [used - produced]
    flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

    %*****************************************************************
    % EXPORTS/IMPORTS
    %*****************************************************************
    % Inorganic Fertilizer imported/exported (negative value is export)
    % Includes land applied + algae growing - locally produced
```

```
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
```

```
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surpassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%*********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
```

```
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%*********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%*********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
```

```
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    end
    % Biomass emission factor tonne/kwh (assumed as wood)
    bitkwh = parause2(129:131) * parause2(132)^-1;
    bitkwh(2) = 0; % Phosphorus corrected to be zero

    %%% Correction. Natural gas, gasoline, and diesel are assumed to
    %%% undergo 100% oxidation, i.e. all C and N is released as emission
    % Natural gas emission factor tonne/kwh (100% oxidation)
    %ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
    % Gasoline emission factor tonne/kwh (100% oxidation)
    %gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
    % Diesel emission factor tonne/kwh (100% oxidation)
    %ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
    % 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
    % Coal combustion generates CCP and emissions, partitioning input materials
    tol_coal_em = zeros(x,3);
    for a = 1:x
        tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
    end
    flowmat(:,219,2:4) = -1 * (tol_coal_em);

    % 220   Emissions from Local NG use (DO, CO, IN, TR)
    % Includes biogas
    flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
    % 221   Emissions from Local Diesel use (DO, CO, IN, TR)
    % includes biodiesel
    flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
    % 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
    flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

    % 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
    %flowmat(:,223,2:4) = -1 * (0);
    % 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
    %flowmat(:,224,2:4) = -1 * (0);
```

```
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
```

```
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%*********************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%*********************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
```

```
end
%************************************************************************
% Water
%************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%************************************************************************
% Nitrogen
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
```

```
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model
```

```
%*********************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%*********************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%*********************************************************************
% Forestry Sector
%*********************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops
```

```
flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%*********************************************************************
% Food Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%*********************************************************************
% Energy Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
```

```
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%*********************************************************************
% Waste Management Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%*********************************************************************
% INDICATOR SCORES ARE CALCULATED                                    *
%*********************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
```

```
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%**************************************************************************
%% NEW indicator
%**************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
```

```
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
```

```
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
```

```
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%*********************************************************************
```

```
% Store data
%************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%************************************************************************
% Estimations associated with land uses
%************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%************************************************************************
% ESTIMATION OF AREA for each designated land use
%************************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
```

```
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water
```

```
% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
```

```
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%************************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%************************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% ************************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ************************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
```

```
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% ************************************************************************
% CALCULATION OF food consumption (by humans)
% ************************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
```

```
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
```

```
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
```

```
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
```

```
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%**********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%**********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%**********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%**********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
```

```
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%*********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%*********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
```

```
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
```

```
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
end
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%**************************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%**************************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
```

```
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
```

```
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                    % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);              % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%*********************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%*********************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
```

```
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
```

```
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
```

```
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
```

```
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    end
    % (b) sewer infiltration index
    sinfil = parause(55) * sinfil;

    % Runoff from pervious areas
    flowmat(:,77,:) = epa - ginfil;

    % Calculation of the fertilization mix (area-wise)
    fertmix

    tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
    tpa = statout(:,27); % total pervious area km2
    %tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
    Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
    % Forested areas accounts also for urban forests
    %*********************************************************************
    % FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
    %*********************************************************************
    % Creates a matrix of the proportion of non-fertilized area for each land
    % use with the size of the alternatives of fertilization
    fractnF = zeros(tfe,size(Areas,2));
    % [grass forestry crops]
    for a = 1:tfe;
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
    end

    % Specification of the fertilizer mix (in terms of area proportion) for
    % each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
    fe_mix = a_mix.* (ones(tfe,3) - fractnF);
    % fe_mix modified. non-fertilized area fraction is added.
    % fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
    fe_mix = cat(1,fractnF(1,:),fe_mix);
    % depth = [number of years]
    fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
    for a = 1:x % each year
        for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
```

```
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%**************************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%**************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);
```

```
% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%************************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%************************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%************************************************************************
% WATER OUTPUT CALCULATIONS
%************************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%************************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%************************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
```

```
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
```

```
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%**********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%**********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
```

```
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
```

```
%*********************************************************************
% Algae growing from WWTP effluent
%*********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
```

```
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%**********************************************************************
% Emissions to air
%**********************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%**********************************************************************
% Activated sludge reactor
%**********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
```

```
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%**************************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%**************************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ====================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
```

```
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%**************************************************************************
% BIOGAS FLOW
%**************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%**************************************************************************
% TREATED SLUDGE FLOW
%**************************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
```

```
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*******************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*******************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
```

```
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%**********************************************************************
% Consistency verification
%**********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));
```

```
% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ***********************************************************
% Calculation of live tree removals
% ***********************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
```

```
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);                % tonne d.m.
if x > 1

%***********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%***********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%***********************************************************************
% CALCULATION OF CARBON POOLS
%***********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
```

```
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
       if a == 1
           statout(a,34) = smcap1(a) + smcap0;
       end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
```

```
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
```

```
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%**********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%**********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%**********************************************************************
% ATMOSPHERIC PROCESSES
%**********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%**********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%**********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
```

```
        b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
        flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
        man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
        flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
        flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
        flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
        b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
        flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
        man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
        flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
        flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
        flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
        b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
        flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
        man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
        flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
        flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
        flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    end
    % manure lbs per head per day (crosschecking)
    % for poultry
    % (flowmat(:,310,6)./statout(:,6))/365 * 2.2
    % Energy content calculated based on Channiwala 2002
    % HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
    % HHV in MJ/kg of dry mass
    % Estimation for H S O Ash is based on their relationship to carbon content
    % in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
    % and A/C = 0.6783
    % Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
    % check O/C = 0.95 approximately
    UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
    Mener   = zeros(3,1); % [cattle poultry swine]
    for a = 0:5:10
        b = a/5;
        Car = parause2(111+a);
        UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
        UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

        % HHV in MJ/kg of dry mass
        Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
            + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
            - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
        % Energy in kWh/y
        flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
        b = a/5;
        Car = parause2(111+a);
        UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
        UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

        % HHV in MJ/kg of dry mass
        Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
            + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
            - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
        % Energy in kWh/y
        flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
        b = a/5;
        Car = parause2(111+a);
        UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
        UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

        % HHV in MJ/kg of dry mass
```

```
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%**********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%**********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
```

```
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%*****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%*****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
```

```
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
     end
     bB = bB + b + 1;
     for b = 1:3 % [N P C]
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
     end
     bB = bB + b + 1;
     for b = 1:3 % [N P C]
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
     end
     bB = bB + b + 1;
end
%*****************************************************************
% Nutrient Content for each type of fertilizer IF SS PL CM SM CP
%*****************************************************************
% First we calculate the water content (moisture) of manure after storage
% & handling (estimated in manure.m) as a fraction w/w of total mass,
% nutrient content as w/w fraction of DM, and energy in kWh/tonne
fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

% Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
% Column of zeros for IF
for a = 1:nmat
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
end
% Now data is corrected to reflect fraction of mass. For compost we use the
% typical compost composition entered by user
% Dry matter in tonne/y
fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
% We convert first column from Water content Mgal/d to w/w of total mass
fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

% Nutrient content corrected to w/w of dry matter
for a = 2:4
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
end
% Energy content corrected to kWh per tonne (fresh matter)
fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);
```

```
% fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
% fraction as from 1 to 5 are.
% Data is added for IF (not calculated above) from user input
% Assumed moisture 0.5%
fm_cont(isnan(fm_cont)) = 0;
fm_cont(:,1,1) = ones(x,1) * 0.005;
fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;
```

```
% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
```

```
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%*************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
```

```
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%*********************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%*********************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
```

```
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*******************************************************************
% MANURE distribution
%*******************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%***********************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%***********************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%*********************************************************************
```

```
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%**************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%**************************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%**************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%**************************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%**************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%**************************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
```

```
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
```

```
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%**********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
```

```
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
```

```
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%**************************************************************************
% CONVERSIONS
%**************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
```

```
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%**********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];
```

```
% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
```

```
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);
```

```
% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  ************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  ************************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
```

```
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    end
    % Total meat produced
    flowmat(:,160,:) = sum(P_meat,2);

    % Total animal (meat) mass accumulated due to increase or decrease of
    % livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
    L_acc = zeros(x,3,nmat);
    % Average livestock growth in tonne/y, based on growth of thousand
    % heads: 'liveg' calculated in 'livepop.m'
    for b = 0:2
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    end
    % [year] x [Cattle Poultry Pigs] x [specie]
    for b = 0:4:8
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    end
    % Total livestock accumulation
    flowmat(:,306,:) = sum(L_acc,2);

    % DAIRY  ************************************************************
    % Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
    % Food Materials (1896) Professor W.O. Atwater). For the total mass
    % will add again the refure mass.
    re_mass = 0.105;
    egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

    % Total dairy mass, milk and eggs in tonne per year
    mil_mass = statout(:,5).* statin(42)* statin(44);
```

```
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ********************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
```

```
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);            % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ***********************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);            % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  **********************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);            % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  ***********************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
```

```
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
```

```
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*********************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*********************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
```

```
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
```

```
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%**********************************************************************
% BIOGAS FLOW
%**********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%**********************************************************************
% FOOD SLUDGE FLOW
%**********************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);
```

```
%**********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%**********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%*********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
```

```
%*********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%*********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%*********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%*********************************************************************
% First part of the energy module. second part is 'energys2.m'
%*********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
```

```
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%**************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%**************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y
```

```
%************************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%************************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%************************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%************************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%************************************************************************
% Estimation of flows for power generation
%************************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
```

```
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%*********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%*********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
```

```
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                           % content in raw coal (dry basis)
                           % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%**************************************************************************
% Emissions from power generation (coal)
%**************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
```

```
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%*********************************************************************
% Algae growing from coal plant emissions
%*********************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%*********************************************************************
% Production of biodiesel
%*********************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
```

```
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
```

```
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%**********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%**********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**********************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%**********************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%**********************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
```

```
%************************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
```

```
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
```

```
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
```

```
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

% Checking: sum(pyFlows(:,1:8,1),2)
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                  % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                  % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                  % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%*********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%*********************************************************************
```

```
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%*********************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%*********************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%*********************************************************************
% LANDFILLING
%*********************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
```

```
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
```

```
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%************************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%************************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%************************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%************************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
```

```
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    end
    % Resulting composted fertilizer - CORRECTING TOTAL MASS
    % Substracting water and nutrient loss
    flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
    % check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
    % Energy content calculated based on Channiwala 2002
    % HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
    % HHV in MJ/kg of dry mass
    % Estimation for H S O Ash is based on their relationship to carbon content
    % in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
    % and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
    % compared to fresh manure ~20-30%. So we double of Singh's values,
    % Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
    % with the ~51% reduction in Carbon content during composting that promotes
    % a larger Ash/C ratio.
    % Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
    % check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
    UAn     = zeros(x,6);    % percentage [C H S O N Ash]
    Drym    = flowmat(:,315,6) - flowmat(:,315,7);
    UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
    UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
    UAn(:,5) = flowmat(:,315,2)./Drym * 100;
    UAn(:,6) = UAn(:,1) * 1.4;
```

```
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%*****************************************************************
% EXPORTS/IMPORTS
%*****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%*****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%*****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
```

```
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surpassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%*********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
```

```
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%*****************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%*****************************************************************
```

```
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
```

```
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
```

```
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
```

```
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%*************************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%*************************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%*************************************************************************
% Water
%*************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%*************************************************************************
% Nitrogen
%*************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%*************************************************************************
% Phosphorus
%*************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
```

```
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
```

```
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%-------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%*************************************************************************
% Rest - no healthy emissions defined
%*************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%*************************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%*************************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%*************************************************************************
% Forestry Sector
%*************************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
```

```
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%**********************************************************************
% Food Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
```

```
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%**********************************************************************
% Energy Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power geneneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%**********************************************************************
% Waste Management Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
```

```
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%*************************************************************************
% INDICATOR SCORES ARE CALCULATED                                        *
%*************************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%************************************************************************
%% NEW indicator
%************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
```

```
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
```

```
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
```

```
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
```

```
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%************************************************************************
% Store data
%************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%************************************************************************
% Estimations associated with land uses
%************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
```

```
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%*********************************************************************
% ESTIMATION OF AREA for each designated land use
%*********************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
```

```
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
```

```
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%************************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%************************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
```

```
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% ************************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ************************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% ************************************************************************
% CALCULATION OF food consumption (by humans)
% ************************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
```

```
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1
```

```
flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
```

```
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
```

```
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%*********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%*********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
```

```
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%*********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%*********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
```

```
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
```

```
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
end
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%*********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%*********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
```

```
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
    end
    % Withdrawals
    flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
    withtot = withtot + flowmat(:,a+61,1);
    % Consumptive use
    flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
    consutot = consutot + flowmat(:,a+64,1);
    % Fuel production capacity time series
    %statout(:,i+21) = inout(:,12+i) + statin(i+22);
    for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
    end
    % Withdrawals
    flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
    withtot = withtot + flowmat(:,a+61,1);
    % Consumptive use
    flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
    consutot = consutot + flowmat(:,a+64,1);
    % Fuel production capacity time series
    %statout(:,i+21) = inout(:,12+i) + statin(i+22);
    for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
    end
    % Withdrawals
    flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
    withtot = withtot + flowmat(:,a+61,1);
    % Consumptive use
    flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
    consutot = consutot + flowmat(:,a+64,1);
    end
    % Total water withdrawal GW + SW
    flowmat(:,70,1) = withtot;

    %Total water withdrawn GW
    flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
    for a = 1:x
        flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
    end
    % Total water withdrawal SW
    flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
    for a = 1:x
        flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
    end
    % Total water withdrawal GW + SW (COMPOSITION)
    flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

    % Total consumptive water use
    flowmat(:,71,1) = -1 * consutot;
    % Total water discharged from fuel production processes
    flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

    %*********************************************************************
    % IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
    %*********************************************************************
```

```
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                   % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);             % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
```

```
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%************************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%************************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];
```

```
        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
```

```
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
```

```
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
end
% (b) sewer infiltration index
sinfil = parause(55) * sinfil;

% Runoff from pervious areas
flowmat(:,77,:) = epa - ginfil;

% Calculation of the fertilization mix (area-wise)
fertmix

tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
tpa = statout(:,27); % total pervious area km2
%tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
% Forested areas accounts also for urban forests
%*********************************************************************
% FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
%*********************************************************************
% Creates a matrix of the proportion of non-fertilized area for each land
% use with the size of the alternatives of fertilization
fractnF = zeros(tfe,size(Areas,2));
% [grass forestry crops]
for a = 1:tfe;
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
end
```

```
% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = a_mix.* (ones(tfe,3) - fractnF);
% fe_mix modified. non-fertilized area fraction is added.
% fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = cat(1,fractnF(1,:),fe_mix);
% depth = [number of years]
fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
for a = 1:x % each year
    for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%************************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
```

```
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%************************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%************************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%************************************************************************
% WATER OUTPUT CALCULATIONS
%************************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);
```

```
%************************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%************************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
```

```
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%*********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%*********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
```

```
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
```

```
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%**********************************************************************
% Algae growing from WWTP effluent
%**********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
```

```
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%************************************************************************
% Emissions to air
%************************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact
```

```
Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%*********************************************************************
% Activated sludge reactor
%*********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%*********************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%*********************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ==================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
```

```
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%**********************************************************************
% BIOGAS FLOW
%**********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%**********************************************************************
% TREATED SLUDGE FLOW
%**********************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
```

```
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*********************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
```

```
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*****************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
```

```
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ************************************************************
% Calculation of live tree removals
% ************************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
```

```
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%***********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%***********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%***********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%***********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%***********************************************************************
% CALCULATION OF CARBON POOLS
%***********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
```

```
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
```

```
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
       if a == 1
           statout(a,34) = smcap1(a) + smcap0;
       end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;
```

```
%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
```

```
%*********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
```

```
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%**********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%**********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];
```

```
% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
```

```
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%*************************************************************
% Recommended Nutrient FOR EACH LAND USE
%*************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
end
%*************************************************************
% Nutrient Content for each type of fertilizer IF SS PL CM SM CP
%*************************************************************
% First we calculate the water content (moisture) of manure after storage
% & handling (estimated in manure.m) as a fraction w/w of total mass,
% nutrient content as w/w fraction of DM, and energy in kWh/tonne
fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

% Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
% Column of zeros for IF
for a = 1:nmat
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
end
```

```
% Now data is corrected to reflect fraction of mass. For compost we use the
% typical compost composition entered by user
% Dry matter in tonne/y
fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
% We convert first column from Water content Mgal/d to w/w of total mass
fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

% Nutrient content corrected to w/w of dry matter
for a = 2:4
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
end
% Energy content corrected to kWh per tonne (fresh matter)
fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

% fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
% fraction as from 1 to 5 are.
% Data is added for IF (not calculated above) from user input
% Assumed moisture 0.5%
fm_cont(isnan(fm_cont)) = 0;
fm_cont(:,1,1) = ones(x,1) * 0.005;
fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);
```

```
% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
```

```
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%**************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
```

```
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%**************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%**************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
```

```
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);
```

```
%************************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%************************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%************************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%************************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%************************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%************************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
```

```
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
```

```
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%*********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
```

```
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
```

```
end
%************************************************************************
%****************** (originally calculated in waters) *******************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%************************************************************************
% CONVERSIONS
%************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
```

```
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%**********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];
```

```
% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;
```

```
% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
```

```
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  ************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  ************************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
```

```
        P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
        P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    end

    % [year] x [Cattle Poultry Pigs] x [specie]
    for b = 0:4:8
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    end
    % Total meat produced
    flowmat(:,160,:) = sum(P_meat,2);

    % Total animal (meat) mass accumulated due to increase or decrease of
    % livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
    L_acc = zeros(x,3,nmat);
    % Average livestock growth in tonne/y, based on growth of thousand
    % heads: 'liveg' calculated in 'livepop.m'
    for b = 0:2
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    end
    % [year] x [Cattle Poultry Pigs] x [specie]
    for b = 0:4:8
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
```

```
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
end
% Total livestock accumulation
flowmat(:,306,:) = sum(L_acc,2);

% DAIRY  ************************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);
```

```
% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);            % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  *************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);            % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ************************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
```

```
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  ************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
```

```
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*********************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*********************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
```

```
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
```

```
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%*********************************************************************
% FOOD SLUDGE FLOW
%*********************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
```

```
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%***********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%***********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
```

```
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%**********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%**********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%**********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%**********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%**********************************************************************
% First part of the energy module. second part is 'energys2.m'
%**********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
```

```
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
```

```
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%*********************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%*********************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%*********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y
```

```
% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%*********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%*********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;
```

```
    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                           % content in raw coal (dry basis)
                           % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;
```

```
flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%************************************************************************
% Algae growing from coal plant emissions
%************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%************************************************************************
% Production of biodiesel
%************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));
```

```
% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%**************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%**************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
```

```
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%*********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%*********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;
```

```
%*************************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%*************************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%*************************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%*************************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%*************************************************************************
```

```
% Preliminary calculations of Oxygen and Hydrogen content
%*********************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
```

```
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
```

```
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

% Checking: sum(pyFlows(:,1:8,1),2)
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;        % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;        % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;        % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                   % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                   % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                   % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
```

```
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%**************************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%**************************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%**************************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%**************************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%**************************************************************************
% LANDFILLING
%**************************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
```

```
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
```

```
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%*********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%*********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%*********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%*********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
```

```
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
end
% Resulting composted fertilizer - CORRECTING TOTAL MASS
% Substracting water and nutrient loss
flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
% check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
```

```
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
% and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
% compared to fresh manure ~20-30%. So we double of Singh's values,
% Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
% with the ~51% reduction in Carbon content during composting that promotes
% a larger Ash/C ratio.
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
UAn     = zeros(x,6);    % percentage [C H S O N Ash]
Drym    = flowmat(:,315,6) - flowmat(:,315,7);
UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = flowmat(:,315,2)./Drym * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
       + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
       - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%****************************************************************
% EXPORTS/IMPORTS
%****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
```

```
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
```

```
%**************************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%**************************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
```

```
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%*********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%*********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF      % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
```

```
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);
```

```
    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
```

```
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%************************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%************************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%************************************************************************
% Water
%************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%************************************************************************
% Nitrogen
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;
```

```
% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
```

```
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%*********************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%*********************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
```

```
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%**********************************************************************
% Forestry Sector
%**********************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%**********************************************************************
% Food Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
```

```
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%**********************************************************************
% Energy Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power geneneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%**********************************************************************
% Waste Management Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
```

```
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%************************************************************************
% INDICATOR SCORES ARE CALCULATED                                       *
%************************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%************************************************************************
%% NEW indicator
%************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
```

```
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
```

```
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
```

```
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
```

```
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%*************************************************************************
% Store data
%*************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%*************************************************************************
% Estimations associated with land uses
%*************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
```

```
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%**********************************************************************
% ESTIMATION OF AREA for each designated land use
%**********************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%**********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%**********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));
```

```
% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
```

```
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
```

```
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%*********************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%*********************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% *********************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% *********************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% *********************************************************************
% CALCULATION OF food consumption (by humans)
% *********************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%*********************************************************************
% FOOD CONSUMPTION
```

```
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
```

```
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
```

```
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);
```

```
%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%*********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%*********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
```

```
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
```

```
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%**********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%**********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
```

```
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    end
    % Correction for air cooling technology for natural gas power generation
    flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

    % Total water withdrawal GW + SW
    flowmat(:,59,1) = withtot;
    %Total water withdrawn GW
    flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
    for a = 1:x
        flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
    end
    % Total water withdrawal SW
    flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
    for a = 1:x
        flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
    end
    % Total water withdrawal GW + SW (COMPOSITION)
    flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

    % Total consumptive water use
```

```
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%**********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%**********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
```

```
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
```

```
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                    % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);              % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%************************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%************************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
```

```
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
```

```
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
```

```
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
end
% (b) sewer infiltration index
sinfil = parause(55) * sinfil;

% Runoff from pervious areas
flowmat(:,77,:) = epa - ginfil;

% Calculation of the fertilization mix (area-wise)
fertmix
```

```
tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
tpa = statout(:,27); % total pervious area km2
%tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
% Forested areas accounts also for urban forests
%*******************************************************************
% FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
%*******************************************************************
% Creates a matrix of the proportion of non-fertilized area for each land
% use with the size of the alternatives of fertilization
fractnF = zeros(tfe,size(Areas,2));
% [grass forestry crops]
for a = 1:tfe;
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
end

% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = a_mix.* (ones(tfe,3) - fractnF);
% fe_mix modified. non-fertilized area fraction is added.
% fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = cat(1,fractnF(1,:),fe_mix);
% depth = [number of years]
fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
for a = 1:x % each year
    for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%*********************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%*********************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;
```

```
% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%*********************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%*********************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                    % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
```

```
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%*********************************************************************
% WATER OUTPUT CALCULATIONS
%*********************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%*********************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%*********************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
```

```
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);
```

```
% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%*********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%*********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
```

```
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%*********************************************************************
% Algae growing from WWTP effluent
%*********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);
```

```
% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%**********************************************************************
% Emissions to air
%**********************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
```

```
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%*********************************************************************
% Activated sludge reactor
%*********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
```

```
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%*********************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%*********************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ====================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
```

```
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%*********************************************************************
% TREATED SLUDGE FLOW
%*********************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
```

```
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*****************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*****************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
```

```
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ***********************************************************
% Calculation of live tree removals
% ***********************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
```

```
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%*********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%*********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%*********************************************************************
% CALCULATION OF CARBON POOLS
%*********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
```

```
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
```

```
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
        if a == 1
            statout(a,34) = smcap1(a) + smcap0;
        end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
```

```
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
```

```
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
%*********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
```

```
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%*********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%*********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
```

```
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
```

```
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%*****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%*****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
end
%*****************************************************************
% Nutrient Content for each type of fertilizer IF SS PL CM SM CP
%*****************************************************************
% First we calculate the water content (moisture) of manure after storage
% & handling (estimated in manure.m) as a fraction w/w of total mass,
% nutrient content as w/w fraction of DM, and energy in kWh/tonne
```

```
fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

% Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
% Column of zeros for IF
for a = 1:nmat
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
end
% Now data is corrected to reflect fraction of mass. For compost we use the
% typical compost composition entered by user
% Dry matter in tonne/y
fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
% We convert first column from Water content Mgal/d to w/w of total mass
fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

% Nutrient content corrected to w/w of dry matter
for a = 2:4
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
end
% Energy content corrected to kWh per tonne (fresh matter)
fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

% fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
% fraction as from 1 to 5 are.
% Data is added for IF (not calculated above) from user input
% Assumed moisture 0.5%
fm_cont(isnan(fm_cont)) = 0;
fm_cont(:,1,1) = ones(x,1) * 0.005;
fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
```

```
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
```

```
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%**************************************************************************
```

```
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%**************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%**************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
```

```
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
```

```
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%************************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%************************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%************************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%************************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%************************************************************************
```

```
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%*********************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%*********************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
```

```
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
```

```
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%**********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
```

```
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%*********************************************************************
%****************** (originally calculated in waters) ******************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%*********************************************************************

 %*********************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%*********************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
```

```
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%**************************************************************************
% CONVERSIONS
%**************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;
```

```
%************************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%************************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
```

```
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
```

```
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
```

```
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*******************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*******************************************************************
% FISH  **********************************************************
flowmat(:,161,:) = 0;      % No significant fish production in the Upper
                            % Chattahoochee Watershed
% MEAT  **********************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
end
% Total meat produced
flowmat(:,160,:) = sum(P_meat,2);

% Total animal (meat) mass accumulated due to increase or decrease of
% livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
L_acc = zeros(x,3,nmat);
% Average livestock growth in tonne/y, based on growth of thousand
% heads: 'liveg' calculated in 'livepop.m'
for b = 0:2
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
end
```

```
% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
end
% Total livestock accumulation
flowmat(:,306,:) = sum(L_acc,2);

% DAIRY  ***********************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
```

```
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);            % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y
```

```
% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);           % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ************************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  *************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;
```

```
% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*************************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*************************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
```

```
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
```

```
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%*********************************************************************
```

```
% FOOD SLUDGE FLOW
%**********************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%**********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%**********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
```

```
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%***********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%***********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%***********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%***********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%************************************************************************
% First part of the energy module. second part is 'energys2.m'
%************************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];
```

```
% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
```

```
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%************************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%************************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%************************************************************************
```

```
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%*********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%*********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%*********************************************************************
% Natural gas and diesel is completed in energy2.m
```

```
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%**********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%**********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                           % content in raw coal (dry basis)
                           % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;
```

```
% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%*********************************************************************
% Algae growing from coal plant emissions
%*********************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);
```

```
    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%**************************************************************************
% Production of biodiesel
%**************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%**************************************************************************
```

```
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%*********************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%*********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%*********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
```

```
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**************************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%**************************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%**************************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%**************************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
```

```
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
```

```
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
```

```
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

% Checking: sum(pyFlows(:,1:8,1),2)
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
```

```
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                   % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                   % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                   % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%************************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%************************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%************************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%************************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%************************************************************************
% LANDFILLING
%************************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
```

```
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
```

```
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%*********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%*********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%*********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%*********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
```

```
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
```

```
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
end
% Resulting composted fertilizer - CORRECTING TOTAL MASS
% Substracting water and nutrient loss
flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
% check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
% and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
% compared to fresh manure ~20-30%. So we double of Singh's values,
% Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
% with the ~51% reduction in Carbon content during composting that promotes
% a larger Ash/C ratio.
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
UAn     = zeros(x,6);    % percentage [C H S O N Ash]
Drym    = flowmat(:,315,6) - flowmat(:,315,7);
UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = flowmat(:,315,2)./Drym * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
       + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
       - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%*****************************************************************
% EXPORTS/IMPORTS
%*****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
```

```
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%*****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%*****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
```

```
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%**********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%**********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
```

```
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%**********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%**********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
```

```
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
```

```
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
```

```
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%*********************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%*********************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%*********************************************************************
% Water
%*********************************************************************
% * Air emission precipitation - evapotranspiration.
```

```
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%************************************************************************
% Nitrogen
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
```

```
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%*********************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%*********************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
```

```
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%**************************************************************************
% Forestry Sector
%**************************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
```

```
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%**********************************************************************
% Food Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%**********************************************************************
% Energy Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereation
```

```
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%**************************************************************************
% Waste Management Sector
%**************************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%**************************************************************************
% INDICATOR SCORES ARE CALCULATED                                         *
%**************************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
```

```
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%**************************************************************************
%% NEW indicator
%**************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
```

```
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
```

```
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
```

```
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%*************************************************************************
% Store data
%*************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
```

```
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%************************************************************************
% Estimations associated with land uses
%************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%************************************************************************
% ESTIMATION OF AREA for each designated land use
%************************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
```

```
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
```

```
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
```

```
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%************************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%************************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% ************************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ************************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
```

```
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% ************************************************************************
% CALCULATION OF food consumption (by humans)
% ************************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
```

```
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
```

```
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);
```

```
%************************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%************************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%************************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%************************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
```

```
end

%*********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%*********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);
```

```
% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%*********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%*********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
```

```
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    end
```

```
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%**************************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%**************************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
```

```
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*************************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*************************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
```

```
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                    % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);              % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%*********************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%*********************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
```

```
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
```

```
        % [water forest grass crops -impervious]
        Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


        % Limit evaporation to the amount of precipitation for
        % [forest grass crops] in mm/year
        for luse_et = 2:4
           if Q_Et(luse_et,4) > precip
               Q_Et(luse_et,4) = precip;
           if Q_Et(luse_et,4) > precip
           if Q_Et(luse_et,4) > precip
           end
        end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);
```

```
% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
```

```
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    end
    % (b) sewer infiltration index
    sinfil = parause(55) * sinfil;

    % Runoff from pervious areas
    flowmat(:,77,:) = epa - ginfil;

    % Calculation of the fertilization mix (area-wise)
    fertmix

    tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
    tpa = statout(:,27); % total pervious area km2
    %tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
    Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
    % Forested areas accounts also for urban forests
    %**********************************************************************
    % FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
    %**********************************************************************
    % Creates a matrix of the proportion of non-fertilized area for each land
    % use with the size of the alternatives of fertilization
    fractnF = zeros(tfe,size(Areas,2));
    % [grass forestry crops]
    for a = 1:tfe;
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
    end

    % Specification of the fertilizer mix (in terms of area proportion) for
    % each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
    fe_mix = a_mix.* (ones(tfe,3) - fractnF);
    % fe_mix modified. non-fertilized area fraction is added.
    % fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
    fe_mix = cat(1,fractnF(1,:),fe_mix);
    % depth = [number of years]
    fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
    for a = 1:x % each year
        for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
```

```
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%************************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
```

```
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%*********************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%*********************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%*********************************************************************
% WATER OUTPUT CALCULATIONS
%*********************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%*********************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%*********************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
```

```
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp
```

```
TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%**********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%**********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
```

```
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%*********************************************************************
% Algae growing from WWTP effluent
%*********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
```

```
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
```

```
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%**************************************************************************
% Emissions to air
%**************************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%**************************************************************************
% Activated sludge reactor
%**************************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
```

```
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%**********************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%**********************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ====================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
```

```
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%************************************************************************
% BIOGAS FLOW
%************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%************************************************************************
% TREATED SLUDGE FLOW
%************************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
```

```
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*******************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*******************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
```

```
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]
```

```
% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ********************************************************
% Calculation of live tree removals
% ********************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%*********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
```

```
%*********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%*********************************************************************
% CALCULATION OF CARBON POOLS
%*********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
```

```
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%**********************************************************************
% CALCULATION of SAWMILLS input and output
%**********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
        if a == 1
            statout(a,34) = smcap1(a) + smcap0;
        end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%**********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%**********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
```

```
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;
```

```
% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
%*********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%*********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%*********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
```

```
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
```

```
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%*********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%*********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
```

```
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
```

```
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
     end
     bB = bB + b + 1;
     for b = 1:3 % [N P C]
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
     end
     bB = bB + b + 1;
 end
 %*****************************************************************
 % Nutrient Content for each type of fertilizer IF SS PL CM SM CP
 %*****************************************************************
 % First we calculate the water content (moisture) of manure after storage
 % & handling (estimated in manure.m) as a fraction w/w of total mass,
 % nutrient content as w/w fraction of DM, and energy in kWh/tonne
 fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

 % Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
 % Column of zeros for IF
 for a = 1:nmat
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
 end
 % Now data is corrected to reflect fraction of mass. For compost we use the
 % typical compost composition entered by user
 % Dry matter in tonne/y
 fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
 % We convert first column from Water content Mgal/d to w/w of total mass
 fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

 % Nutrient content corrected to w/w of dry matter
 for a = 2:4
     fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
     fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
     fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
 end
 % Energy content corrected to kWh per tonne (fresh matter)
 fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

 % fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
 % fraction as from 1 to 5 are.
 % Data is added for IF (not calculated above) from user input
 % Assumed moisture 0.5%
 fm_cont(isnan(fm_cont)) = 0;
 fm_cont(:,1,1) = ones(x,1) * 0.005;
 fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;
```

```
% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated
```

```
% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
```

```
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
```

```
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
```

```
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%*************************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%*************************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%**********************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);
```

```
% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%*********************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%*********************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%*********************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%*********************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%*********************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
```

```
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
```

```
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%**********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
```

```
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;
```

```
%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%**************************************************************************
% CONVERSIONS
%**************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
```

```
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%**********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
```

```
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
```

```
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
```

```
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  ************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  ************************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
```

```
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    end
    % Total meat produced
    flowmat(:,160,:) = sum(P_meat,2);

    % Total animal (meat) mass accumulated due to increase or decrease of
    % livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
    L_acc = zeros(x,3,nmat);
    % Average livestock growth in tonne/y, based on growth of thousand
    % heads: 'liveg' calculated in 'livepop.m'
    for b = 0:2
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    end
    % [year] x [Cattle Poultry Pigs] x [specie]
    for b = 0:4:8
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    end
    % Total livestock accumulation
    flowmat(:,306,:) = sum(L_acc,2);

    % DAIRY  ***********************************************************
    % Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
    % Food Materials (1896) Professor W.O. Atwater). For the total mass
    % will add again the refure mass.
    re_mass = 0.105;
    egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

    % Total dairy mass, milk and eggs in tonne per year
    mil_mass = statout(:,5).* statin(42)* statin(44);
    egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

    d_mass = [egg_mass mil_mass];
    % Matrix for diary production
    P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

    % For correlation between carbohydrates, protein, fat, nitrogen, carbon,
```

```
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ********************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);            % tonnes/y
```

```
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);           % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ***********************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  ************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
```

```
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively
```

```
% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%**********************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%**********************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);
```

```
% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%**********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%**********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%**********************************************************************
% Waste reducing methods
%**********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%**********************************************************************
% Anaerobic digestion of food waste
%**********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
```

```
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%**********************************************************************
% BIOGAS FLOW
%**********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%**********************************************************************
% FOOD SLUDGE FLOW
%**********************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%**********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%**********************************************************************
mswinci

AshC = 0.02;
```

```
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%**********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%**********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
```

```
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%*********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%*********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%*********************************************************************
% First part of the energy module. second part is 'energys2.m'
%*********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
```

```
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%************************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%************************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
```

```
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%*********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
```

```
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%*********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%*********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
```

```
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                           % content in raw coal (dry basis)
                           % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%*********************************************************************
% Algae growing from coal plant emissions
%*********************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
```

```
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%************************************************************************
% Production of biodiesel
%************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
```

```
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
```

```
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%**********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%**********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**********************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%**********************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%**********************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%**********************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
      m_H_pl = 0;
      m_O_pl = 0;
```

```
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%**************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%**************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
```

```
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
```

```
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);
```

```
% Checking: sum(pyFlows(:,1:8,1),2)
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                  % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                  % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                  % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%**********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%**********************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%**********************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%**********************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);
```

```
%************************************************************************
% LANDFILLING
%************************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
```

```
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
```

```
end
%************************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%************************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%************************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%************************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
```

```
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    end
    % Resulting composted fertilizer - CORRECTING TOTAL MASS
    % Substracting water and nutrient loss
    flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
    % check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
    % Energy content calculated based on Channiwala 2002
    % HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
    % HHV in MJ/kg of dry mass
    % Estimation for H S O Ash is based on their relationship to carbon content
    % in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
    % and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
    % compared to fresh manure ~20-30%. So we double of Singh's values,
    % Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
    % with the ~51% reduction in Carbon content during composting that promotes
    % a larger Ash/C ratio.
    % Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
    % check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
    UAn     = zeros(x,6);    % percentage [C H S O N Ash]
    Drym    = flowmat(:,315,6) - flowmat(:,315,7);
    UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
    UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
    UAn(:,5) = flowmat(:,315,2)./Drym * 100;
    UAn(:,6) = UAn(:,1) * 1.4;
    UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

    %Energy in MJ/kg
    comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
           + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
           - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);
```

```
% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%*****************************************************************
% EXPORTS/IMPORTS
%*****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%*****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%*****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
```

```
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%*********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
```

```
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%*********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%*********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
```

```
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
   tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);
```

```
% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
```

```
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%***********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%***********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
```

```
%************************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%************************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%************************************************************************
% Water
%************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%************************************************************************
% Nitrogen
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
```

```
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
```

```
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%************************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%************************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%************************************************************************
% Forestry Sector
%************************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
```

```
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%*********************************************************************
% Food Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%*********************************************************************
```

```
% Energy Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%*********************************************************************
% Waste Management Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);
```

```
%*************************************************************************
% INDICATOR SCORES ARE CALCULATED                                        *
%*************************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%*************************************************************************
%% NEW indicator
%*************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
```

```
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
```

```
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
```

```
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
```

```
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%************************************************************************
% Store data
%************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%************************************************************************
% Estimations associated with land uses
%************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
```

```
    emat(2) = 0;
end
%*********************************************************************
% ESTIMATION OF AREA for each designated land use
%*********************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
```

```
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
```

```
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%************************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%************************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
```

```
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% ************************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ************************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
       flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
           - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% ************************************************************************
% CALCULATION OF food consumption (by humans)
% ************************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
```

```
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
```

```
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*************************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*************************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
```

```
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
```

```
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%**********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%**********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
```

```
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%**********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%**********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%**********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%**********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
```

```
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
```

```
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
end
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%*********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%*********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
```

```
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;
```

```
% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                   % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);             % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%*********************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%*********************************************************************
```

```
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
```

```
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];
```

```
% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;
```

```
% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
end
% (b) sewer infiltration index
sinfil = parause(55) * sinfil;

% Runoff from pervious areas
flowmat(:,77,:) = epa - ginfil;

% Calculation of the fertilization mix (area-wise)
fertmix

tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
tpa = statout(:,27); % total pervious area km2
%tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
% Forested areas accounts also for urban forests
%*******************************************************************
% FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
%*******************************************************************
% Creates a matrix of the proportion of non-fertilized area for each land
% use with the size of the alternatives of fertilization
fractnF = zeros(tfe,size(Areas,2));
% [grass forestry crops]
for a = 1:tfe;
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
end

% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = a_mix.* (ones(tfe,3) - fractnF);
% fe_mix modified. non-fertilized area fraction is added.
% fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = cat(1,fractnF(1,:),fe_mix);
```

```
% depth = [number of years]
fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
for a = 1:x % each year
    for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%************************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
```

```
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%**********************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%**********************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%**********************************************************************
% WATER OUTPUT CALCULATIONS
%**********************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%**********************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%**********************************************************************
```

```
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
```

```
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%*********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%*********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
```

```
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
```

```
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%**********************************************************************
% Algae growing from WWTP effluent
%**********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
```

```
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%************************************************************************
% Emissions to air
%************************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;
```

```
%*********************************************************************
% Activated sludge reactor
%*********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%*********************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%*********************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ====================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
```

```
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%*********************************************************************
% TREATED SLUDGE FLOW
%*********************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
```

```
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*********************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
           parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
           parause(227) * log(infBOD) + ...
           parause(228) * log(effBOD) + ...
           parause(236) * log(100 * (1 - parause(64))) + ...
           parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);
```

```
% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*****************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
```

```
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ************************************************************
% Calculation of live tree removals
% ************************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*****************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*****************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
```

```
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%***********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%***********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%***********************************************************************
% CALCULATION OF CARBON POOLS
%***********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
```

```
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%**********************************************************************
% CALCULATION of SAWMILLS input and output
%**********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
        if a == 1
            statout(a,34) = smcap1(a) + smcap0;
        end
end
```

```
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
```

```
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
%*********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert
```

```
man_m = [parause2(113) parause2(118) parause2(123)]/100;

%*********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%*********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
```

```
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%**********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%**********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
```

```
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);
```

```
%****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
end
%****************************************************************
% Nutrient Content for each type of fertilizer IF SS PL CM SM CP
%****************************************************************
% First we calculate the water content (moisture) of manure after storage
% & handling (estimated in manure.m) as a fraction w/w of total mass,
% nutrient content as w/w fraction of DM, and energy in kWh/tonne
fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

% Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
% Column of zeros for IF
for a = 1:nmat
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
end
% Now data is corrected to reflect fraction of mass. For compost we use the
% typical compost composition entered by user
% Dry matter in tonne/y
fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
% We convert first column from Water content Mgal/d to w/w of total mass
fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);
```

```
% Nutrient content corrected to w/w of dry matter
for a = 2:4
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
end
% Energy content corrected to kWh per tonne (fresh matter)
fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

% fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
% fraction as from 1 to 5 are.
% Data is added for IF (not calculated above) from user input
% Assumed moisture 0.5%
fm_cont(isnan(fm_cont)) = 0;
fm_cont(:,1,1) = ones(x,1) * 0.005;
fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
```

```
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
```

```
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
```

```
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%**************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%**************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
```

```
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%*********************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%*********************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
```

```
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%**************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%**************************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%**************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%**************************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%**************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%**************************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%**************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%**************************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year
```

```
% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;
```

```
% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%*********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
```

```
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%**************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
```

```
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%************************************************************************
% CONVERSIONS
%************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner
```

```
En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%**********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
```

```
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
```

```
        DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
        DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    end
    % Consistency check. Feed intake kg per head daily.
    % cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
    % poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
    % swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
    for b = 0:5:10 % [cattle poultry swine]
        a = b/5 + 1;
        flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
        flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
        flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
            (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
        flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
        flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
        flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
        flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
        a = b/5 + 1;
        flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
        flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
        flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
            (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
        flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
        flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
        flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
        flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
        a = b/5 + 1;
        flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
        flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
        flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
            (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
        flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
        flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
        flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
        flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    end
    % We correct the energy calculated by digestible energy (DE) and converted
    % to gross energy which is assumed to be 75%
    flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

    % Protein and phosphorus content is corrected to reach minimum requirements
    % Based on the National Research Council a minimum of 15% CP and 0.6% P are
    % required for Poultry (NRC 1961 in WPS Journal).
    % All requirements follow National Research Council recommendations.
    % Equations were derived from, see 'livestock and farm inventory.xls':
    % Nutrient Requirements of Swine, NRC 2000
    % Minimum requirements for all livestock population, tonne/y
    minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
    % DE in Mcal/y and rate in kg/kcal
    % Cattle (beef and dairy)
    minN(:,1,1) = 7.84688E-06 * DE(:,1);
    minN(:,1,2) = 1.19415E-06 * DE(:,1);
    % Poultry
    minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
    minN(:,2,2) = DM_all(:,2) * 0.006;
    % Swine
    if BW(3) < 20
    else
        Pnit = -1.178E-04 * BW(3) + 3.276E-02;
        Ppho = -1.535E-05 * BW(3) + 5.524E-03;
```

```
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  ************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  ************************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
```

```
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    end
    % Total meat produced
    flowmat(:,160,:) = sum(P_meat,2);

    % Total animal (meat) mass accumulated due to increase or decrease of
    % livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
    L_acc = zeros(x,3,nmat);
    % Average livestock growth in tonne/y, based on growth of thousand
    % heads: 'liveg' calculated in 'livepop.m'
    for b = 0:2
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    end
    % [year] x [Cattle Poultry Pigs] x [specie]
    for b = 0:4:8
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    end
    % Total livestock accumulation
    flowmat(:,306,:) = sum(L_acc,2);
```

```
% DAIRY  ************************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
```

```
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);           % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);           % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ***********************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
```

```
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  ************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
```

```
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*********************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*********************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
```

```
% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
```

```
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%**************************************************************************
% BIOGAS FLOW
%**************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%**************************************************************************
% FOOD SLUDGE FLOW
%**************************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
```

```
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%*********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%*********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
```

```
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%*********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%*********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%*********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%*********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%*********************************************************************
% First part of the energy module. second part is 'energys2.m'
%*********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
```

```
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
```

```
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%*********************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%*********************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%*********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
```

```
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%*********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%*********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
```

```
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                            % content in raw coal (dry basis)
                            % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);
```

```
% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%************************************************************************
% Algae growing from coal plant emissions
%************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%************************************************************************
% Production of biodiesel
%************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
```

```
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%**************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%**************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
```

```
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%**********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%**********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**********************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%**********************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
```

```
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%************************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%************************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
```

```
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;
```

```
% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
```

```
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
    % Energy in kWh/y
       pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
    end
    %Check energy by mass partition
    % [condensate solids gas]
    %El = pyenergy(:,2) * pyYield(1,10);
    %Ec = pyenergy(:,3) * pyYield(2,10);
    %Eg = pyenergy(:,4) * pyYield(3,10);
    %El + Ec + Eg;
    %alternative for calculating mass of each phase (using mass yield factors)
    % [condensate solids gas]
    pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
    pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
    pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

    % Checking: sum(pyFlows(:,1:8,1),2)
    % sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
    % pyFlows(:,10,1) + py_feed(:,:,7)
    % Pyrolysis outputs (in flowmat format)
    % Water content in mass units tonne/y (assumed to be lost during the
    % pyrolysis and upgrading process)
    flowmat(:,265,7)    = 0;                    % Liquid phase
    flowmat(:,267,7)    = 0;                    % Char phase
    flowmat(:,266,7)    = 0;                    % gas phase
    % Water content in volume units Mgal/d
    flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
    flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
    flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
    %N, P, and C in mass units tonne/y
    for a = 2:4
        flowmat(:,265,a) = pyFlows(:,a,2);                  % Liquid phase
        flowmat(:,267,a) = pyFlows(:,a,3);                  % Char phase
        flowmat(:,266,a) = pyFlows(:,a,4);                  % gas phase
        flowmat(:,265,a) = pyFlows(:,a,2);                  % Liquid phase
        flowmat(:,267,a) = pyFlows(:,a,3);                  % Char phase
        flowmat(:,266,a) = pyFlows(:,a,4);                  % gas phase
        flowmat(:,265,a) = pyFlows(:,a,2);                  % Liquid phase
        flowmat(:,267,a) = pyFlows(:,a,3);                  % Char phase
        flowmat(:,266,a) = pyFlows(:,a,4);                  % gas phase
    end
    % Total mass tonne/y, Liquid, char, gas (NO water)
    flowmat(:,265,6)    = pyFlows(:,10,2);
    flowmat(:,267,6)    = pyFlows(:,10,3);
    flowmat(:,266,6)    = pyFlows(:,10,4);
    % Energy flow kwh/y
    flowmat(:,265,5)    = pyFlows(:,9,2);                    % Liquid phase
    flowmat(:,267,5)    = pyFlows(:,9,3);                    % Char phase
    flowmat(:,266,5)    = pyFlows(:,9,4);                    % gas phase

    % check total mass
    %sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
    % Emissions from the pyrolysis and upgrading process
    % All the water in feed
    %flowmat(:,303,7) = flowmat(:,268,7);
    flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
    flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

    % Energy usage by the pyrolysis process (electricity terms) based on Reed
    % XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
```

```
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%**********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%**********************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%**********************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%**********************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%**********************************************************************
% LANDFILLING
%**********************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];
```

```
% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
```

```
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%*********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%*********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%*********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%*********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
```

```
    CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
    if any(CtoN < 10 | CtoN > 30) == true
        end
    end
    for a = 1:nmat
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    end
    % Resulting composted fertilizer - CORRECTING TOTAL MASS
    % Substracting water and nutrient loss
    flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
    % check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
    % Energy content calculated based on Channiwala 2002
    % HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
    % HHV in MJ/kg of dry mass
    % Estimation for H S O Ash is based on their relationship to carbon content
    % in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
    % and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
    % compared to fresh manure ~20-30%. So we double of Singh's values,
    % Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
    % with the ~51% reduction in Carbon content during composting that promotes
```

```
% a larger Ash/C ratio.
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
UAn     = zeros(x,6);    % percentage [C H S O N Ash]
Drym    = flowmat(:,315,6) - flowmat(:,315,7);
UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = flowmat(:,315,2)./Drym * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
       + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
       - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%****************************************************************
% EXPORTS/IMPORTS
%****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%****************************************************************
```

```
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%*********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
```

```
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
```

```
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%*********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%*********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero
```

```
%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
```

```
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
```

```
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%*************************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%*************************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%*************************************************************************
% Water
%*************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%*************************************************************************
% Nitrogen
%*************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);
```

```
%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
```

```
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%************************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%************************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);
```

```
%**************************************************************************
% Forestry Sector
%**************************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%**************************************************************************
% Food Sector
%**************************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;
```

```
% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%**********************************************************************
% Energy Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%**********************************************************************
% Waste Management Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;
```

```
% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%**************************************************************************
% INDICATOR SCORES ARE CALCULATED                                         *
%**************************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%**************************************************************************
%% NEW indicator
%**************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
```

```
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
```

```
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
```

```
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim
```

```
caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%************************************************************************
% Store data
%************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%************************************************************************
% Estimations associated with land uses
%************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));
```

```
%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%**********************************************************************
% ESTIMATION OF AREA for each designated land use
%**********************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%**********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%**********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
```

```
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
```

```
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
```

```
end
%************************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%************************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% ************************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ************************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% ************************************************************************
% CALCULATION OF food consumption (by humans)
% ************************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
```

```
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
```

```
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
```

```
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
```

```
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%**************************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%**************************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
```

```
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
```

```
%*********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%*********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
```

```
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
end
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%*********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%*********************************************************************
```

```
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);
```

```
% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
```

```
ruia = parause(57) * (precip - imwt);                    % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);              % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%************************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%************************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]
```

```
        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2
```

```
% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
```

```
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
end
% (b) sewer infiltration index
sinfil = parause(55) * sinfil;

% Runoff from pervious areas
flowmat(:,77,:) = epa - ginfil;

% Calculation of the fertilization mix (area-wise)
fertmix

tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
tpa = statout(:,27); % total pervious area km2
%tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
% Forested areas accounts also for urban forests
%*********************************************************************
% FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
%*********************************************************************
% Creates a matrix of the proportion of non-fertilized area for each land
```

```
% use with the size of the alternatives of fertilization
fractnF = zeros(tfe,size(Areas,2));
% [grass forestry crops]
for a = 1:tfe;
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
end

% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = a_mix.* (ones(tfe,3) - fractnF);
% fe_mix modified. non-fertilized area fraction is added.
% fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = cat(1,fractnF(1,:),fe_mix);
% depth = [number of years]
fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
for a = 1:x % each year
    for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%***********************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%***********************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y
```

```
% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%**********************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%**********************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%**********************************************************************
% WATER OUTPUT CALCULATIONS
```

```
%************************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%************************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%************************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
```

```
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
```

```
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%**********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%**********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
```

```
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%*********************************************************************
% Algae growing from WWTP effluent
%*********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
```

```
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%*********************************************************************
% Emissions to air
%*********************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
```

```
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%*********************************************************************
% Activated sludge reactor
%*********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%*********************************************************************
```

```
% Solid waste - SEWAGE SLUDGE GENERATION
%*********************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ====================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%=================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%*********************************************************************
```

```
% TREATED SLUDGE FLOW
%*********************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*******************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
```

```
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*******************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
```

```
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ********************************************************
% Calculation of live tree removals
% ********************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
```

```
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%*********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%*********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%*********************************************************************
% CALCULATION OF CARBON POOLS
%*********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
```

```
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;
```

```
% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
        if a == 1
            statout(a,34) = smcap1(a) + smcap0;
        end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
```

```
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
```

```
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%**********************************************************************
% ATMOSPHERIC PROCESSES
%**********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%*********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%*********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
```

```
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%*********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%*********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
```

```
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
```

```
        % HHV in MJ/kg of dry mass
        Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
            UAn(a,3) - 0.1034 * UAn(a,4)...
            - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
        % Energy in kWh/y
        flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    end
    % Losses from manure storage and handling losses
    flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
    % No energy content in emissions
    EneLoss = flowmat(:,294,5); % Variable of lost energy
    flowmat(:,294,5) = 0;

    % TOTAL Manure Generated after handling and storage losses
    % [cattle poultry swine] added
    flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

    %*****************************************************************
    % Recommended Nutrient FOR EACH LAND USE
    %*****************************************************************
    % Recommended Nutrient per land use (in tonne/km2)
    % Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
    % Pre-allocating
    rec_fert = zeros(3);
    bB = 92; % Initializing
    for a = 1:3 % [grass forestry crops]
        for b = 1:3 % [N P C]
           rec_fert(a,b) = parause2(bB + b)./10;
           rec_fert(a,b) = parause2(bB + b)./10;
           rec_fert(a,b) = parause2(bB + b)./10;
        end
        bB = bB + b + 1;
        for b = 1:3 % [N P C]
           rec_fert(a,b) = parause2(bB + b)./10;
           rec_fert(a,b) = parause2(bB + b)./10;
           rec_fert(a,b) = parause2(bB + b)./10;
        end
        bB = bB + b + 1;
        for b = 1:3 % [N P C]
           rec_fert(a,b) = parause2(bB + b)./10;
           rec_fert(a,b) = parause2(bB + b)./10;
           rec_fert(a,b) = parause2(bB + b)./10;
        end
        bB = bB + b + 1;
    end
    %*****************************************************************
    % Nutrient Content for each type of fertilizer IF SS PL CM SM CP
    %*****************************************************************
    % First we calculate the water content (moisture) of manure after storage
    % & handling (estimated in manure.m) as a fraction w/w of total mass,
    % nutrient content as w/w fraction of DM, and energy in kWh/tonne
    fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

    % Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
    % Column of zeros for IF
    for a = 1:nmat
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
```

```
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    end
    % Now data is corrected to reflect fraction of mass. For compost we use the
    % typical compost composition entered by user
    % Dry matter in tonne/y
    fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
    % We convert first column from Water content Mgal/d to w/w of total mass
    fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

    % Nutrient content corrected to w/w of dry matter
    for a = 2:4
        fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
        fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
        fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    end
    % Energy content corrected to kWh per tonne (fresh matter)
    fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

    % fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
    % fraction as from 1 to 5 are.
    % Data is added for IF (not calculated above) from user input
    % Assumed moisture 0.5%
    fm_cont(isnan(fm_cont)) = 0;
    fm_cont(:,1,1) = ones(x,1) * 0.005;
    fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

    % Data of ideal compost is added from user input
    fm_cont(:,6,1) = parause2(128)/100;
    fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
    % energy calculated from ultimate analysis
    % Energy in kWh per tonne of total mass (wet basis)
    EneComp % fm_cont(:,6,5) calculated

    UAn     = zeros(x,6);    % percentage [C H S O N Ash]

    UAn(:,1) = fm_cont(:,6,4) * 100;
    UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
    UAn(:,5) = fm_cont(:,6,2) * 100;
    UAn(:,6) = UAn(:,1) * 1.4;
    UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

    %Energy in MJ/kg
    comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
            + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
            - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

    % Energy in kWh per tonne of total mass (wet basis)
    fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

    %*****************************************************************
    % NUTRIENT APPLICATION RATE
    %*****************************************************************
```

```
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
```

```
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
```

```
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%**************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%**************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
```

```
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
```

```
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%************************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%************************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%************************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%************************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%************************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%************************************************************************
```

```
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
```

```
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%*****************************************************************
```

```
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%*********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
```

```
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%************************************************************************
% CONVERSIONS
%************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
```

```
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%**********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
```

```
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
```

```
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
```

```
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*****************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
```

```
%*****************************************************************
% FISH  *********************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  *********************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
end
% Total meat produced
flowmat(:,160,:) = sum(P_meat,2);

% Total animal (meat) mass accumulated due to increase or decrease of
% livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
L_acc = zeros(x,3,nmat);
% Average livestock growth in tonne/y, based on growth of thousand
% heads: 'liveg' calculated in 'livepop.m'
for b = 0:2
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
end
% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
```

```
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
end
% Total livestock accumulation
flowmat(:,306,:) = sum(L_acc,2);

% DAIRY  ************************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);
```

```
% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);            % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);            % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d
```

```
% VEGGIES  ************************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  ************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
```

```
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% -------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%--------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*************************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*************************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);
```

```
% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
```

```
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%*********************************************************************
% FOOD SLUDGE FLOW
%*********************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
```

```
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%***********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%***********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
```

```
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%*********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%*********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%*********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%*********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%**********************************************************************
% First part of the energy module. second part is 'energys2.m'
%**********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
```

```
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));
```

```
%************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%************************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%************************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%************************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%************************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
```

```
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%*********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%*********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix
```

```
% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                            % content in raw coal (dry basis)
                            % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;
```

```
% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%************************************************************************
% Algae growing from coal plant emissions
%************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;
```

```
%************************************************************************
% Production of biodiesel
%************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);
```

```
% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%*********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%*********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
```

```
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**************************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%**************************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%**************************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%**************************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
```

```
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
```

```
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
```

```
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

% Checking: sum(pyFlows(:,1:8,1),2)
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
```

```
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                    % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                    % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                    % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%**********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%**********************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%**********************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%**********************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%**********************************************************************
% LANDFILLING
%**********************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
```

```
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;
```

```
% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%*********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%*********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%*********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%*********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
```

```
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
```

```
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
end
% Resulting composted fertilizer - CORRECTING TOTAL MASS
% Substracting water and nutrient loss
flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
% check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
% and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
% compared to fresh manure ~20-30%. So we double of Singh's values,
% Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
% with the ~51% reduction in Carbon content during composting that promotes
% a larger Ash/C ratio.
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
UAn     = zeros(x,6);    % percentage [C H S O N Ash]
Drym    = flowmat(:,315,6) - flowmat(:,315,7);
UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = flowmat(:,315,2)./Drym * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%****************************************************************
% EXPORTS/IMPORTS
%****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
```

```
%*****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
```

```
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surpassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%*********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
```

```
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%**********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%**********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
```

```
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE
```

```
for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
```

```
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%*********************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%*********************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%*************************************************************************
% Water
%*************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%*********************************************************************
% Nitrogen
```

```
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
```

```
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%************************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%************************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;
```

```
pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Forestry Sector
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Food Sector
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;
```

```
I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%**************************************************************************
% Energy Sector
%**************************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
```

```
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%*********************************************************************
% Waste Management Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%*********************************************************************
% INDICATOR SCORES ARE CALCULATED                                    *
%*********************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
```

```
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%************************************************************************
%% NEW indicator
%************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
```

```
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;
```

```
      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)
```

```
      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%************************************************************************
% Store data
%************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%************************************************************************
% Estimations associated with land uses
```

```
%************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%************************************************************************
% ESTIMATION OF AREA for each designated land use
%************************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%************************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
```

```
%**************************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%**************************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%**************************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
```

```
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
```

```
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%**********************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%**********************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% **********************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% **********************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
       flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
           - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% **********************************************************************
% CALCULATION OF food consumption (by humans)
% **********************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
```

```
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%**************************************************************************
% FOOD CONSUMPTION
%**************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
```

```
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
```

```
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
```

```
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%*********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%*********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);
```

```
% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
```

```
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%*********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%*********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
```

```
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    end
    % Correction for air cooling technology for natural gas power generation
    flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

    % Total water withdrawal GW + SW
    flowmat(:,59,1) = withtot;
    %Total water withdrawn GW
    flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
    for a = 1:x
        flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
```

```
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%**************************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%**************************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
```

```
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
```

```
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                    % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);              % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%************************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%************************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;
```

```
        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
```

```
            if Q_Et(luse_et,4) > precip
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
```

```
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
end
% (b) sewer infiltration index
sinfil = parause(55) * sinfil;
```

```
% Runoff from pervious areas
flowmat(:,77,:) = epa - ginfil;

% Calculation of the fertilization mix (area-wise)
fertmix

tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
tpa = statout(:,27); % total pervious area km2
%tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
% Forested areas accounts also for urban forests
%**********************************************************************
% FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
%**********************************************************************
% Creates a matrix of the proportion of non-fertilized area for each land
% use with the size of the alternatives of fertilization
fractnF = zeros(tfe,size(Areas,2));
% [grass forestry crops]
for a = 1:tfe;
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
end

% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = a_mix.* (ones(tfe,3) - fractnF);
% fe_mix modified. non-fertilized area fraction is added.
% fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = cat(1,fractnF(1,:),fe_mix);
% depth = [number of years]
fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
for a = 1:x % each year
    for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%************************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
```

```
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%*********************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%*********************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                    % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
```

```
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%*********************************************************************
% WATER OUTPUT CALCULATIONS
%*********************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%*********************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%*********************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
```

```
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;
```

```
% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%*********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%*********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
```

```
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%*********************************************************************
% Algae growing from WWTP effluent
%*********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
```

```
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%*********************************************************************
% Emissions to air
%*********************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
```

```
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%**********************************************************************
% Activated sludge reactor
%**********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);
```

```
if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%**************************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%**************************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ====================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
```

```
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%************************************************************************
% BIOGAS FLOW
%************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%************************************************************************
% TREATED SLUDGE FLOW
%************************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
```

```
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*************************************************************************
% ENERGY USAGE BY WWTP
%*************************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*******************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*******************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
```

```
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% *********************************************************
```

```
% Calculation of live tree removals
% ********************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%*********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%*********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%*********************************************************************
% CALCULATION OF CARBON POOLS
%*********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));
```

```
fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
```

```
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
        if a == 1
            statout(a,34) = smcap1(a) + smcap0;
        end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
```

```
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
```

```
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%**********************************************************************
% ATMOSPHERIC PROCESSES
%**********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%**********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%**********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
```

```
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%*******************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%*******************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
```

```
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
```

```
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%*****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%*****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
```

```
end
%*****************************************************************
% Nutrient Content for each type of fertilizer IF SS PL CM SM CP
%*****************************************************************
% First we calculate the water content (moisture) of manure after storage
% & handling (estimated in manure.m) as a fraction w/w of total mass,
% nutrient content as w/w fraction of DM, and energy in kWh/tonne
fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

% Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
% Column of zeros for IF
for a = 1:nmat
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
end
% Now data is corrected to reflect fraction of mass. For compost we use the
% typical compost composition entered by user
% Dry matter in tonne/y
fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
% We convert first column from Water content Mgal/d to w/w of total mass
fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

% Nutrient content corrected to w/w of dry matter
for a = 2:4
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
end
% Energy content corrected to kWh per tonne (fresh matter)
fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

% fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
% fraction as from 1 to 5 are.
% Data is added for IF (not calculated above) from user input
% Assumed moisture 0.5%
fm_cont(isnan(fm_cont)) = 0;
fm_cont(:,1,1) = ones(x,1) * 0.005;
fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]
```

```
UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
       + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
       - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
```

```
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
```

```
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
```

```
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);
```

```
for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%**************************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%**************************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%***********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%***********************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%***********************************************************************
```

```
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%*********************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%*********************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%*********************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%*********************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
```

```
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
```

```
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%**********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
```

```
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));
```

```
% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%**************************************************************************
% CONVERSIONS
%**************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
```

```
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%**********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
```

```
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
```

```
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;
```

```
% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  ***********************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  ***********************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
end
% Total meat produced
flowmat(:,160,:) = sum(P_meat,2);

% Total animal (meat) mass accumulated due to increase or decrease of
% livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
```

```
L_acc = zeros(x,3,nmat);
% Average livestock growth in tonne/y, based on growth of thousand
% heads: 'liveg' calculated in 'livepop.m'
for b = 0:2
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
end
% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
end
% Total livestock accumulation
flowmat(:,306,:) = sum(L_acc,2);

% DAIRY  ***********************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
```

```
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);            % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
```

```
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);           % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ************************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  *************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);
```

```
% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*****************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
```

```
%*********************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
```

```
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
```

```
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%**********************************************************************
% FOOD SLUDGE FLOW
%**********************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%**********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%**********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
```

```
        flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
    % Emission from food products in MSW incineration
        flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
    % Emission from sewage sludge in incineration
        flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
    % Emission from wood products in MSW incineration
        flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
    % Emission from food products in MSW incineration
        flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
    % Emission from sewage sludge in incineration
        flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
    % Emission from wood products in MSW incineration
        flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
    % Emission from food products in MSW incineration
        flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
    % Emission from sewage sludge in incineration
        flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
    % Emission from wood products in MSW incineration
        flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
    % Emission from food products in MSW incineration
        flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
    % Emission from sewage sludge in incineration
        flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
    % Emission from wood products in MSW incineration
        flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
    % Emission from food products in MSW incineration
        flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
    % Emission from sewage sludge in incineration
        flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
    % Emission from wood products in MSW incineration
        flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
    % Emission from food products in MSW incineration
        flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
    % Emission from sewage sludge in incineration
        flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
    end
    % Energy content of incineration gases, complete
    flowmat(:,143,5) = 0;
    flowmat(:,280,5) = 0;
    flowmat(:,336,5) = 0;

    %*********************************************************************
    % RECOVERY OF ENERGY FROM MSW COMBUSTION
    %*********************************************************************
    % Energy generated is calculated based on a the energy content of materials
    % and a combustion efficiency
    % Energy from MSW (wood + food + sludge)
    % Based on efficiency
    flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
    % Only sludge
    flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

    %*********************************************************************
    % CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
    %*********************************************************************
    % For this we use the content of volatile components in
    MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
    MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
    flowmat(:,276,:) = MSWfeed + MSWemission;
```

```
%************************************************************************
% First part of the energy module. second part is 'energys2.m'
%************************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;
```

```
% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%************************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%************************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
```

```
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%*********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
```

```
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%*********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%*********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
```

```
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                           % content in raw coal (dry basis)
                           % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%************************************************************************
% Algae growing from coal plant emissions
%************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
```

```
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%************************************************************************
% Production of biodiesel
%************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
```

```
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%**************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%**************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
```

```
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%*********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%*********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%*********************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%*********************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%*********************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%*********************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
```

```
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
```

```
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
```

```
        pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
            + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
            - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
    % Energy in kWh/y
        pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
    % Dry mass based on ultimate composition
        pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
    % Ultimate composition on a dry basis (percentage)
        pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
            pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
    % HHV in MJ/kg of dry mass
        pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
            + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
            - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
    % Energy in kWh/y
        pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
    % Dry mass based on ultimate composition
        pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
    % Ultimate composition on a dry basis (percentage)
        pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
            pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
    % HHV in MJ/kg of dry mass
        pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
            + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
            - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
    % Energy in kWh/y
        pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
    % Dry mass based on ultimate composition
        pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
    % Ultimate composition on a dry basis (percentage)
        pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
            pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
    % HHV in MJ/kg of dry mass
        pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
            + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
            - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
    % Energy in kWh/y
        pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
    end
    %Check energy by mass partition
    % [condensate solids gas]
    %El = pyenergy(:,2) * pyYield(1,10);
    %Ec = pyenergy(:,3) * pyYield(2,10);
    %Eg = pyenergy(:,4) * pyYield(3,10);
    %El + Ec + Eg;
    %alternative for calculating mass of each phase (using mass yield factors)
    % [condensate solids gas]
    pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
    pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
    pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

    % Checking: sum(pyFlows(:,1:8,1),2)
    % sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
    % pyFlows(:,10,1) + py_feed(:,:,7)
    % Pyrolysis outputs (in flowmat format)
    % Water content in mass units tonne/y (assumed to be lost during the
    % pyrolysis and upgrading process)
    flowmat(:,265,7)    = 0;                    % Liquid phase
    flowmat(:,267,7)    = 0;                    % Char phase
    flowmat(:,266,7)    = 0;                    % gas phase
    % Water content in volume units Mgal/d
```

```
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                  % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                  % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                  % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                  % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                  % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                  % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                  % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                  % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                  % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                     % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                     % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                     % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%**********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%**********************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%**********************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%**********************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%**********************************************************************
% LANDFILLING
%**********************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
```

```
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne
```

```
LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%*********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%*********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);
```

```
%*********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%*********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
```

```
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    end
    % Resulting composted fertilizer - CORRECTING TOTAL MASS
    % Substracting water and nutrient loss
    flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
    % check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
    % Energy content calculated based on Channiwala 2002
    % HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
    % HHV in MJ/kg of dry mass
    % Estimation for H S O Ash is based on their relationship to carbon content
    % in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
    % and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
    % compared to fresh manure ~20-30%. So we double of Singh's values,
    % Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
    % with the ~51% reduction in Carbon content during composting that promotes
    % a larger Ash/C ratio.
    % Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
    % check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
    UAn     = zeros(x,6);    % percentage [C H S O N Ash]
    Drym    = flowmat(:,315,6) - flowmat(:,315,7);
    UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
    UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
    UAn(:,5) = flowmat(:,315,2)./Drym * 100;
    UAn(:,6) = UAn(:,1) * 1.4;
    UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

    %Energy in MJ/kg
    comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
           + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
           - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

    % Energy in kWh/y
    flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

    % Exported or imported organic fertilizer. Includes composted material and
    % solids from food waste anaerobic digestion
    % [used - produced]
    flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

    %*****************************************************************
    % EXPORTS/IMPORTS
```

```
%*****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%*****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%*****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
```

```
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%**********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%**********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
```

```
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%*********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%*********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
```

```
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
```

```
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);
```

```
% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%*********************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%*********************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE
```

```
Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%************************************************************************
% Water
%************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%************************************************************************
% Nitrogen
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
```

```
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
```

```
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%**********************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%**********************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%**********************************************************************
% Forestry Sector
%**********************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
```

```
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%*********************************************************************
% Food Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%*********************************************************************
% Energy Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
```

```
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%**********************************************************************
% Waste Management Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%**********************************************************************
% INDICATOR SCORES ARE CALCULATED                                      *
%**********************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
```

```
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%************************************************************************
%% NEW indicator
%************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
```

```
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
```

```
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
```

```
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
```

```
%
%*************************************************************************
% Store data
%*************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%*************************************************************************
% Estimations associated with land uses
%*************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%*************************************************************************
% ESTIMATION OF AREA for each designated land use
%*************************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
```

```
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-----------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
```

```
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
```

```
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%************************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%************************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% ************************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ************************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
```

```
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% ************************************************************************
% CALCULATION OF food consumption (by humans)
% ************************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
```

```
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
```

```
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*************************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*************************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
```

```
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%************************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%************************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%************************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%************************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
```

```
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%**********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%**********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%**********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%**********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
```

```
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%*********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%*********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
```

```
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
```

```
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
end
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%**********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%**********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
```

```
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
```

```
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                   % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);             % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%*********************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%*********************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
```

```
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
```

```
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
```

```
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
```

```
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    end
    % (b) sewer infiltration index
    sinfil = parause(55) * sinfil;

    % Runoff from pervious areas
    flowmat(:,77,:) = epa - ginfil;

    % Calculation of the fertilization mix (area-wise)
    fertmix

    tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
    tpa = statout(:,27); % total pervious area km2
    %tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
    Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
    % Forested areas accounts also for urban forests
    %*********************************************************************
    % FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
    %*********************************************************************
    % Creates a matrix of the proportion of non-fertilized area for each land
    % use with the size of the alternatives of fertilization
    fractnF = zeros(tfe,size(Areas,2));
    % [grass forestry crops]
    for a = 1:tfe;
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
    end

    % Specification of the fertilizer mix (in terms of area proportion) for
    % each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
    fe_mix = a_mix.* (ones(tfe,3) - fractnF);
    % fe_mix modified. non-fertilized area fraction is added.
    % fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
    fe_mix = cat(1,fractnF(1,:),fe_mix);
    % depth = [number of years]
    fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
    for a = 1:x % each year
        for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
```

```
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%************************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);
```

```
% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%************************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%************************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%************************************************************************
% WATER OUTPUT CALCULATIONS
%************************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%************************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%************************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
```

```
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
```

```
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%*********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%*********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
```

```
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
```

```
%**************************************************************************
% Algae growing from WWTP effluent
%**************************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
```

```
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%*********************************************************************
% Emissions to air
%*********************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%*********************************************************************
% Activated sludge reactor
%*********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
```

```
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%**************************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%**************************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ====================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
```

```
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%************************************************************************
% BIOGAS FLOW
%************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%************************************************************************
% TREATED SLUDGE FLOW
%************************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
```

```
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*******************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*******************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
```

```
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%**************************************************************************
% Consistency verification
%**************************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));
```

```
% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ***********************************************************
% Calculation of live tree removals
% ***********************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
```

```
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);                % tonne d.m.
if x > 1

%*********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%*********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%*********************************************************************
% CALCULATION OF CARBON POOLS
%*********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
```

```
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
        if a == 1
            statout(a,34) = smcap1(a) + smcap0;
        end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;
```

```
%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
```

```
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
%*********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%*********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%*********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
```

```
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
```

```
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%*********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%*********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
```

```
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%*****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%*****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
```

```
            rec_fert(a,b) = parause2(bB + b)./10;
            rec_fert(a,b) = parause2(bB + b)./10;
            rec_fert(a,b) = parause2(bB + b)./10;
        end
        bB = bB + b + 1;
        for b = 1:3 % [N P C]
            rec_fert(a,b) = parause2(bB + b)./10;
            rec_fert(a,b) = parause2(bB + b)./10;
            rec_fert(a,b) = parause2(bB + b)./10;
        end
        bB = bB + b + 1;
        for b = 1:3 % [N P C]
            rec_fert(a,b) = parause2(bB + b)./10;
            rec_fert(a,b) = parause2(bB + b)./10;
            rec_fert(a,b) = parause2(bB + b)./10;
        end
        bB = bB + b + 1;
    end
    %*****************************************************************
    % Nutrient Content for each type of fertilizer IF SS PL CM SM CP
    %*****************************************************************
    % First we calculate the water content (moisture) of manure after storage
    % & handling (estimated in manure.m) as a fraction w/w of total mass,
    % nutrient content as w/w fraction of DM, and energy in kWh/tonne
    fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

    % Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
    % Column of zeros for IF
    for a = 1:nmat
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    end
    % Now data is corrected to reflect fraction of mass. For compost we use the
    % typical compost composition entered by user
    % Dry matter in tonne/y
    fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
    % We convert first column from Water content Mgal/d to w/w of total mass
    fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

    % Nutrient content corrected to w/w of dry matter
    for a = 2:4
        fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
        fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
        fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    end
    % Energy content corrected to kWh per tonne (fresh matter)
    fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

    % fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
```

```
% fraction as from 1 to 5 are.
% Data is added for IF (not calculated above) from user input
% Assumed moisture 0.5%
fm_cont(isnan(fm_cont)) = 0;
fm_cont(:,1,1) = ones(x,1) * 0.005;
fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;
```

```
% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
```

```
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%*************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
```

```
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
```

```
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*******************************************************************
% MANURE distribution
%*******************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%***********************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%***********************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%*********************************************************************
```

```
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%*********************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%*********************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%*********************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%*********************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%*********************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
```

```
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
```

```
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%**********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
```

```
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
```

```
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%**************************************************************************
% CONVERSIONS
%**************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
```

```
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%**********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];
```

```
% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
```

```
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);
```

```
% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  ************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  ************************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
```

```
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    end
    % Total meat produced
    flowmat(:,160,:) = sum(P_meat,2);

    % Total animal (meat) mass accumulated due to increase or decrease of
    % livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
    L_acc = zeros(x,3,nmat);
    % Average livestock growth in tonne/y, based on growth of thousand
    % heads: 'liveg' calculated in 'livepop.m'
    for b = 0:2
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    end
    % [year] x [Cattle Poultry Pigs] x [specie]
    for b = 0:4:8
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    end
    % Total livestock accumulation
    flowmat(:,306,:) = sum(L_acc,2);

    % DAIRY  ************************************************************
    % Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
    % Food Materials (1896) Professor W.O. Atwater). For the total mass
    % will add again the refure mass.
    re_mass = 0.105;
    egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

    % Total dairy mass, milk and eggs in tonne per year
    mil_mass = statout(:,5).* statin(42)* statin(44);
```

```
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ********************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
```

```
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);           % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ***********************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);           % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ***********************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  ***********************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
```

```
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
```

```
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*************************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*************************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
```

```
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
```

```
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%**************************************************************************
% BIOGAS FLOW
%**************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%**************************************************************************
% FOOD SLUDGE FLOW
%**************************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);
```

```
%*********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%*********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%********************************************************************
```

```
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%*********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%*********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%*********************************************************************
% First part of the energy module. second part is 'energys2.m'
%*********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
```

```
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%************************************************************************
```

```
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%*********************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%*********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
```

```
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%*********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%*********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
ccpcalc
```

```
ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                           % content in raw coal (dry basis)
                           % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
```

```
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%*********************************************************************
% Algae growing from coal plant emissions
%*********************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%*********************************************************************
% Production of biodiesel
%*********************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
```

```
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
```

```
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%**************************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%**************************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**************************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%**************************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%**************************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%**************************************************************************
```

```
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%*************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%*************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
```

```
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);
```

```
% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
```

```
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

% Checking: sum(pyFlows(:,1:8,1),2)
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                   % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                   % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                   % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%*********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%*********************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);
```

```
%**************************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%**************************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%**************************************************************************
% LANDFILLING
%**************************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
```

```
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
```

```
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%***********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%***********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%***********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%***********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
```

```
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    end
    % Resulting composted fertilizer - CORRECTING TOTAL MASS
    % Substracting water and nutrient loss
    flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
    % check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
    % Energy content calculated based on Channiwala 2002
    % HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
    % HHV in MJ/kg of dry mass
    % Estimation for H S O Ash is based on their relationship to carbon content
    % in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
    % and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
    % compared to fresh manure ~20-30%. So we double of Singh's values,
    % Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
    % with the ~51% reduction in Carbon content during composting that promotes
    % a larger Ash/C ratio.
    % Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
    % check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
    UAn     = zeros(x,6);    % percentage [C H S O N Ash]
    Drym    = flowmat(:,315,6) - flowmat(:,315,7);
    UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
    UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
    UAn(:,5) = flowmat(:,315,2)./Drym * 100;
    UAn(:,6) = UAn(:,1) * 1.4;
    UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));
```

```
%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%*****************************************************************
% EXPORTS/IMPORTS
%*****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%*****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%*****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
```

```
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%*********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
```

```
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%*****************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%*****************************************************************
```

```
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
```

```
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
```

```
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;
```

```
% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%************************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%************************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%************************************************************************
% Water
%************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%************************************************************************
% Nitrogen
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
```

```
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%-----------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
```

```
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%************************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%************************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%************************************************************************
% Forestry Sector
%************************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
```

```
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%**************************************************************************
% Food Sector
%**************************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)
```

```
% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%*********************************************************************
% Energy Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%*********************************************************************
% Waste Management Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
```

```
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%************************************************************************
% INDICATOR SCORES ARE CALCULATED                                       *
%************************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%************************************************************************
%% NEW indicator
%************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
```

```
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
```

```
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
```

```
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
```

```
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%************************************************************************
% Store data
%************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%************************************************************************
% Estimations associated with land uses
%************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
```

```
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%*********************************************************************
% ESTIMATION OF AREA for each designated land use
%*********************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
```

```
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);
```

```
% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%*********************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%*********************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
```

```
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% ************************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ************************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% ************************************************************************
% CALCULATION OF food consumption (by humans)
% ************************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
```

```
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1
```

```
flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);
```

```
% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
```

```
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%**************************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%**************************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;
```

```
% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%*********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%*********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
```

```
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
```

```
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
end
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%*********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%*********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
```

```
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
    end
    % Withdrawals
    flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
    withtot = withtot + flowmat(:,a+61,1);
    % Consumptive use
    flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
    consutot = consutot + flowmat(:,a+64,1);
    % Fuel production capacity time series
    %statout(:,i+21) = inout(:,12+i) + statin(i+22);
    for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
    end
    % Withdrawals
    flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
    withtot = withtot + flowmat(:,a+61,1);
    % Consumptive use
    flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
    consutot = consutot + flowmat(:,a+64,1);
    % Fuel production capacity time series
    %statout(:,i+21) = inout(:,12+i) + statin(i+22);
    for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
    end
    % Withdrawals
    flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
    withtot = withtot + flowmat(:,a+61,1);
    % Consumptive use
    flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
    consutot = consutot + flowmat(:,a+64,1);
    end
    % Total water withdrawal GW + SW
    flowmat(:,70,1) = withtot;

    %Total water withdrawn GW
    flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
    for a = 1:x
        flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
    end
    % Total water withdrawal SW
    flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
    for a = 1:x
        flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
    end
    % Total water withdrawal GW + SW (COMPOSITION)
    flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

    % Total consumptive water use
    flowmat(:,71,1) = -1 * consutot;
    % Total water discharged from fuel production processes
    flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

    %*********************************************************************
    % IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
    %*********************************************************************
    % Calculates the factor that relates low intensity urban areas to
```

```
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%**********************************************************************
% WATER INPUTS ESTIMATION
%**********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                          % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                    % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);              % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
```

```
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%************************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%************************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];
```

```
        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
```

```
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
```

```
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
end
% (b) sewer infiltration index
sinfil = parause(55) * sinfil;

% Runoff from pervious areas
flowmat(:,77,:) = epa - ginfil;

% Calculation of the fertilization mix (area-wise)
fertmix

tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
tpa = statout(:,27); % total pervious area km2
%tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
% Forested areas accounts also for urban forests
%*********************************************************************
% FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
%*********************************************************************
% Creates a matrix of the proportion of non-fertilized area for each land
% use with the size of the alternatives of fertilization
fractnF = zeros(tfe,size(Areas,2));
% [grass forestry crops]
for a = 1:tfe;
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
end
```

```
% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = a_mix.* (ones(tfe,3) - fractnF);
% fe_mix modified. non-fertilized area fraction is added.
% fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = cat(1,fractnF(1,:),fe_mix);
% depth = [number of years]
fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
for a = 1:x % each year
    for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%************************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
```

```
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%*********************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%*********************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%*********************************************************************
% WATER OUTPUT CALCULATIONS
%*********************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);
```

```
%************************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%************************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
```

```
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%*********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%*********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
```

```
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
```

```
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%**********************************************************************
% Algae growing from WWTP effluent
%**********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
```

```
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%************************************************************************
% Emissions to air
%************************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
```

```
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%*********************************************************************
% Activated sludge reactor
%*********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%*********************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%*********************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ==================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
```

```
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%**********************************************************************
% BIOGAS FLOW
%**********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%**********************************************************************
% TREATED SLUDGE FLOW
%**********************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
```

```
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*********************************************************************
% ENERGY USAGE BY WWTP
%*********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*******************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
```

```
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*****************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
```

```
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ***********************************************************
% Calculation of live tree removals
% ***********************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
```

```
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%*********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%*********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%*********************************************************************
% CALCULATION OF CARBON POOLS
%*********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
```

```
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
```

```
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
       if a == 1
           statout(a,34) = smcap1(a) + smcap0;
       end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
```

```
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
```

```
%**********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%*********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%*********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
```

```
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%**********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%**********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];
```

```
% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
```

```
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%*****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%*****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
end
%*****************************************************************
% Nutrient Content for each type of fertilizer IF SS PL CM SM CP
%*****************************************************************
% First we calculate the water content (moisture) of manure after storage
% & handling (estimated in manure.m) as a fraction w/w of total mass,
% nutrient content as w/w fraction of DM, and energy in kWh/tonne
fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

% Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
% Column of zeros for IF
for a = 1:nmat
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
end
```

```
% Now data is corrected to reflect fraction of mass. For compost we use the
% typical compost composition entered by user
% Dry matter in tonne/y
fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
% We convert first column from Water content Mgal/d to w/w of total mass
fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

% Nutrient content corrected to w/w of dry matter
for a = 2:4
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
end
% Energy content corrected to kWh per tonne (fresh matter)
fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

% fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
% fraction as from 1 to 5 are.
% Data is added for IF (not calculated above) from user input
% Assumed moisture 0.5%
fm_cont(isnan(fm_cont)) = 0;
fm_cont(:,1,1) = ones(x,1) * 0.005;
fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
       + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
       - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
```

```
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
```

```
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
```

```
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%**************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%**************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
```

```
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);
```

```
%************************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%************************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%************************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%************************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%************************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%************************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
```

```
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
```

```
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%*********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
```

```
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
```

```
%************************************************************************
%****************** (originally calculated in waters) *******************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%************************************************************************
% CONVERSIONS
%************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
```

```
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%**********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];
```

```
% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;
```

```
% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
```

```
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  ************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  ************************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
```

```
        P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
        P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    end

    % [year] x [Cattle Poultry Pigs] x [specie]
    for b = 0:4:8
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
        a = (b/4)+1;
        P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
        P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
        P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
        P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    end
    % Total meat produced
    flowmat(:,160,:) = sum(P_meat,2);

    % Total animal (meat) mass accumulated due to increase or decrease of
    % livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
    L_acc = zeros(x,3,nmat);
    % Average livestock growth in tonne/y, based on growth of thousand
    % heads: 'liveg' calculated in 'livepop.m'
    for b = 0:2
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    end
    % [year] x [Cattle Poultry Pigs] x [specie]
    for b = 0:4:8
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
```

```
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
end
% Total livestock accumulation
flowmat(:,306,:) = sum(L_acc,2);

% DAIRY  ************************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);
```

```
% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);           % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  *************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);           % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ************************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a
```

```
% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  ************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
```

```
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*********************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*********************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
```

```
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%**********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%**********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%**********************************************************************
% Waste reducing methods
%**********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%**********************************************************************
% Anaerobic digestion of food waste
```

```
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%*********************************************************************
% FOOD SLUDGE FLOW
%*********************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
```

```
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%***********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%***********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
```

```
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%*********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%*********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%*********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%*********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%*********************************************************************
% First part of the energy module. second part is 'energys2.m'
%*********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
```

```
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%**************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%**************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
```

```
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%**********************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%**********************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%**********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%**********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
```

```
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%*********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%*********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;
```

```
    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                           % content in raw coal (dry basis)
                           % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;
```

```
flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%************************************************************************
% Algae growing from coal plant emissions
%************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%************************************************************************
% Production of biodiesel
%************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));
```

```
% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%**************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%**************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
```

```
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%*********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%*********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;
```

```
%*************************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%*************************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%*************************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%*************************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%*************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
```

```
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
```

```
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
   pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
   pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
```

```
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
  % Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
  % HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
  % Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
  end
  %Check energy by mass partition
  % [condensate solids gas]
  %El = pyenergy(:,2) * pyYield(1,10);
  %Ec = pyenergy(:,3) * pyYield(2,10);
  %Eg = pyenergy(:,4) * pyYield(3,10);
  %El + Ec + Eg;
  %alternative for calculating mass of each phase (using mass yield factors)
  % [condensate solids gas]
  pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
  pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
  pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

  % Checking: sum(pyFlows(:,1:8,1),2)
  % sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
  % pyFlows(:,10,1) + py_feed(:,:,7)
  % Pyrolysis outputs (in flowmat format)
  % Water content in mass units tonne/y (assumed to be lost during the
  % pyrolysis and upgrading process)
  flowmat(:,265,7)    = 0;                    % Liquid phase
  flowmat(:,267,7)    = 0;                    % Char phase
  flowmat(:,266,7)    = 0;                    % gas phase
  % Water content in volume units Mgal/d
  flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
  flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
  flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
  %N, P, and C in mass units tonne/y
  for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
  end
  % Total mass tonne/y, Liquid, char, gas (NO water)
  flowmat(:,265,6)    = pyFlows(:,10,2);
  flowmat(:,267,6)    = pyFlows(:,10,3);
  flowmat(:,266,6)    = pyFlows(:,10,4);
  % Energy flow kwh/y
  flowmat(:,265,5)    = pyFlows(:,9,2);                    % Liquid phase
  flowmat(:,267,5)    = pyFlows(:,9,3);                    % Char phase
  flowmat(:,266,5)    = pyFlows(:,9,4);                    % gas phase

  % check total mass
  %sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
  % Emissions from the pyrolysis and upgrading process
```

```
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%**************************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%**************************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%**************************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%**************************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%**************************************************************************
% LANDFILLING
%**************************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
```

```
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
```

```
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%*********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%*********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%*********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%*********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);
```

```
% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
end
% Resulting composted fertilizer - CORRECTING TOTAL MASS
% Substracting water and nutrient loss
flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
% check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
```

```
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
% and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
% compared to fresh manure ~20-30%. So we double of Singh's values,
% Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
% with the ~51% reduction in Carbon content during composting that promotes
% a larger Ash/C ratio.
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
UAn     = zeros(x,6);    % percentage [C H S O N Ash]
Drym    = flowmat(:,315,6) - flowmat(:,315,7);
UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = flowmat(:,315,2)./Drym * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
       + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
       - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%*****************************************************************
% EXPORTS/IMPORTS
%*****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%*****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%*****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
```

```
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
```

```
%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%*********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
```

```
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%**********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%**********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
```

```
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
```

```
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
```

```
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%*************************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%*************************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%*************************************************************************
% Water
%*************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%*************************************************************************
% Nitrogen
%*************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
```

```
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
```

```
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%*********************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%*********************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
```

```
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%*********************************************************************
% Forestry Sector
%*********************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%*********************************************************************
% Food Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
```

```
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%**********************************************************************
% Energy Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power geneneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%**********************************************************************
% Waste Management Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;
```

```
% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%**************************************************************************
% INDICATOR SCORES ARE CALCULATED                                         *
%**************************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%**************************************************************************
%% NEW indicator
%**************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
```

```
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
```

```
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
```

```
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';
```

```
      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%************************************************************************
% Store data
%************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%************************************************************************
% Estimations associated with land uses
%************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
```

```
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%************************************************************************
% ESTIMATION OF AREA for each designated land use
%************************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%************************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%************************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
```

```
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
```

```
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d
```

```
% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%*********************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%*********************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% *********************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% *********************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% *********************************************************************
% CALCULATION OF food consumption (by humans)
% *********************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%*********************************************************************
% FOOD CONSUMPTION
```

```
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
```

```
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
```

```
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
```

```
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%*********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%*********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
```

```
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
```

```
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%**********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%**********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
```

```
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
end
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
```

```
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%*****************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%*****************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
       if b == 1
           statout(b,a+21) = inout(b,12+a) + statin(a+22);
       end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
```

```
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%**********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%**********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%**********************************************************************
% WATER INPUTS ESTIMATION
%**********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;
```

```
% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                    % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);              % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%************************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%************************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
```

```
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);
```

```
if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
```

```
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
end
% (b) sewer infiltration index
sinfil = parause(55) * sinfil;

% Runoff from pervious areas
flowmat(:,77,:) = epa - ginfil;

% Calculation of the fertilization mix (area-wise)
fertmix

tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
```

```
tpa = statout(:,27); % total pervious area km2
%tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
% Forested areas accounts also for urban forests
%*********************************************************************
% FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
%*********************************************************************
% Creates a matrix of the proportion of non-fertilized area for each land
% use with the size of the alternatives of fertilization
fractnF = zeros(tfe,size(Areas,2));
% [grass forestry crops]
for a = 1:tfe;
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
end

% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = a_mix.* (ones(tfe,3) - fractnF);
% fe_mix modified. non-fertilized area fraction is added.
% fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = cat(1,fractnF(1,:),fe_mix);
% depth = [number of years]
fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
for a = 1:x % each year
    for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%************************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;
```

```
% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%*********************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%*********************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                    % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
```

```
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%*********************************************************************
% WATER OUTPUT CALCULATIONS
%*********************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%*********************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%*********************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
```

```
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
```

```
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%*********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%*********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
```

```
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%*********************************************************************
% Algae growing from WWTP effluent
%*********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);
```

```
% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%*********************************************************************
% Emissions to air
%*********************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
```

```
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%**************************************************************************
% Activated sludge reactor
%**************************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
```

```
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%*********************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%*********************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ====================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
```

```
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%**********************************************************************
% TREATED SLUDGE FLOW
%**********************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%**********************************************************************
% ENERGY USAGE BY WWTP
%**********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
```

```
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*****************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*****************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
```

```
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ************************************************************
% Calculation of live tree removals
% ************************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
```

```
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%*********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%*********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%*********************************************************************
% CALCULATION OF CARBON POOLS
%*********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
```

```
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
```

```
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
        if a == 1
            statout(a,34) = smcap1(a) + smcap0;
        end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
```

```
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
```

```
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
%*********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
```

```
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%*********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%*********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
```

```
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
```

```
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
end
%****************************************************************
% Nutrient Content for each type of fertilizer IF SS PL CM SM CP
%****************************************************************
% First we calculate the water content (moisture) of manure after storage
% & handling (estimated in manure.m) as a fraction w/w of total mass,
% nutrient content as w/w fraction of DM, and energy in kWh/tonne
fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species
```

```
% Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
% Column of zeros for IF
for a = 1:nmat
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
end
% Now data is corrected to reflect fraction of mass. For compost we use the
% typical compost composition entered by user
% Dry matter in tonne/y
fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
% We convert first column from Water content Mgal/d to w/w of total mass
fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

% Nutrient content corrected to w/w of dry matter
for a = 2:4
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
end
% Energy content corrected to kWh per tonne (fresh matter)
fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

% fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
% fraction as from 1 to 5 are.
% Data is added for IF (not calculated above) from user input
% Assumed moisture 0.5%
fm_cont(isnan(fm_cont)) = 0;
fm_cont(:,1,1) = ones(x,1) * 0.005;
fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
```

```
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%****************************************************************
% NUTRIENT APPLICATION RATE
%****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
```

```
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%*********************************************************************
```

```
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%*********************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%*********************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
```

```
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
```

```
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%************************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%************************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%************************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%************************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
```

```
%************************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%************************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
```

```
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
```

```
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%**************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%**************************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
```

```
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
```

```
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%**************************************************************************
% CONVERSIONS
%**************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;
```

```
%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%*********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;
```

```
% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
```

```
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
```

```
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*******************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*******************************************************************
% FISH  ***********************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  ***********************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
end
% Total meat produced
flowmat(:,160,:) = sum(P_meat,2);

% Total animal (meat) mass accumulated due to increase or decrease of
% livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
L_acc = zeros(x,3,nmat);
% Average livestock growth in tonne/y, based on growth of thousand
% heads: 'liveg' calculated in 'livepop.m'
for b = 0:2
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
end
```

```
% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
end
% Total livestock accumulation
flowmat(:,306,:) = sum(L_acc,2);

% DAIRY  ************************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
```

```
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);           % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y
```

```
% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);           % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ************************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  *************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
```

```
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*************************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*************************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
```

```
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
```

```
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%*********************************************************************
```

```
% FOOD SLUDGE FLOW
%*********************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%*********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%*********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
```

```
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
  % Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
  % Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
  % Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
  % Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
  % Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
  % Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
  % Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
  % Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
  % Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
  % Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
  % Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
  % Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
  % Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%**********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%**********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%**********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%**********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%***********************************************************************
% First part of the energy module. second part is 'energys2.m'
%***********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];
```

```
% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));
```

```
% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%************************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%************************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%************************************************************************
```

```
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%*********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%*********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%*********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
```

```
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                           % content in raw coal (dry basis)
                           % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;
```

```
% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%************************************************************************
% Algae growing from coal plant emissions
%************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
```

```
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%************************************************************************
% Production of biodiesel
%************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%************************************************************************
```

```
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%*********************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%*********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%*********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
```

```
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**************************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%**************************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%**************************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%**************************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
```

```
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
```

```
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
```

```
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

% Checking: sum(pyFlows(:,1:8,1),2)
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
```

```
    flowmat(:,265,a) = pyFlows(:,a,2);                  % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                  % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                  % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                  % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                  % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                  % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                    % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                    % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                    % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%**********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%**********************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%**********************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%**********************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%**********************************************************************
% LANDFILLING
%**********************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
```

```
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
```

```
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%*********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%*********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%*********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%*********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
```

```
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
```

```
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
end
% Resulting composted fertilizer - CORRECTING TOTAL MASS
% Substracting water and nutrient loss
flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
% check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
% and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
% compared to fresh manure ~20-30%. So we double of Singh's values,
% Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
% with the ~51% reduction in Carbon content during composting that promotes
% a larger Ash/C ratio.
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
UAn     = zeros(x,6);    % percentage [C H S O N Ash]
Drym    = flowmat(:,315,6) - flowmat(:,315,7);
UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = flowmat(:,315,2)./Drym * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%*****************************************************************
% EXPORTS/IMPORTS
%*****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
```

```
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%*****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%*****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
```

```
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%*********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
```

```
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%**********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%**********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
```

```
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
```

```
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
```

```
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%*********************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%*********************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%*********************************************************************
% Water
%*********************************************************************
% * Air emission precipitation - evapotranspiration.
```

```
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%************************************************************************
% Nitrogen
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
```

```
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%-----------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%-----------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%*********************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%*********************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
```

```
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%**********************************************************************
% Forestry Sector
%**********************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);
```

```
%**************************************************************************
% Food Sector
%**************************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%**************************************************************************
% Energy Sector
%**************************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereration
```

```
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%*********************************************************************
% Waste Management Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%*********************************************************************
% INDICATOR SCORES ARE CALCULATED                                    *
%*********************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
```

```
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%************************************************************************
%% NEW indicator
%************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
```

```
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
```

```
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
```

```
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%************************************************************************
% Store data
%************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
```

```
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%************************************************************************
% Estimations associated with land uses
%************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%************************************************************************
% ESTIMATION OF AREA for each designated land use
%************************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
```

```
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%-----------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
```

```
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
```

```
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%************************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%************************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% ************************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ************************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;
```

```
% ************************************************************************
% CALCULATION OF food consumption (by humans)
% ************************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
```

```
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
```

```
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*************************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*************************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);
```

```
%**********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%**********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%**********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%**********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end
```

```
%*********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%*********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);
```

```
% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%**************************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%**************************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
```

```
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    end
```

```
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%**********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%**********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
```

```
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%**********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%**********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
```

```
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                    % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);              % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%*********************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%*********************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;
```

```
% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
```

```
        % [water forest grass crops -impervious]
        Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


        % Limit evaporation to the amount of precipitation for
        % [forest grass crops] in mm/year
        for luse_et = 2:4
           if Q_Et(luse_et,4) > precip
           if Q_Et(luse_et,4) > precip
           if Q_Et(luse_et,4) > precip
           end
        end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
```

```
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
```

```
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
        l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
        sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    end
    % (b) sewer infiltration index
    sinfil = parause(55) * sinfil;

    % Runoff from pervious areas
    flowmat(:,77,:) = epa - ginfil;

    % Calculation of the fertilization mix (area-wise)
    fertmix

    tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
    tpa = statout(:,27); % total pervious area km2
    %tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
    Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
    % Forested areas accounts also for urban forests
    %*******************************************************************
    % FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
    %*******************************************************************
    % Creates a matrix of the proportion of non-fertilized area for each land
    % use with the size of the alternatives of fertilization
    fractnF = zeros(tfe,size(Areas,2));
    % [grass forestry crops]
    for a = 1:tfe;
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
       fractnF(a,:) = (1-parause(130:132)/100);
    end

    % Specification of the fertilizer mix (in terms of area proportion) for
    % each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
    fe_mix = a_mix.* (ones(tfe,3) - fractnF);
    % fe_mix modified. non-fertilized area fraction is added.
    % fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
    fe_mix = cat(1,fractnF(1,:),fe_mix);
    % depth = [number of years]
    fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
    for a = 1:x % each year
        for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
             % Mix of fertilizers as Area km2 [grass forestry crops]
            fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
```

```
%************************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%************************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
```

```
flowmat(:,321,:) = sinfil;

%************************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%************************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%************************************************************************
% WATER OUTPUT CALCULATIONS
%************************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%************************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%************************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
```

```
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp
```

```
TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%**********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%**********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
```

```
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%*********************************************************************
% Algae growing from WWTP effluent
%*********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
```

```
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
```

```
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%*********************************************************************
% Emissions to air
%*********************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%*********************************************************************
% Activated sludge reactor
%*********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
```

```
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%**************************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%**************************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ====================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;
```

```
% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%************************************************************************
% BIOGAS FLOW
%************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%************************************************************************
% TREATED SLUDGE FLOW
%************************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
```

```
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%*************************************************************************
% ENERGY USAGE BY WWTP
%*************************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*******************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
           parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
           parause(227) * log(infBOD) + ...
           parause(228) * log(effBOD) + ...
           parause(236) * log(100 * (1 - parause(64))) + ...
           parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*******************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
       parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
       parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
       parause(238) * log(statin(57)) + ...
       parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
```

```
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
```

```
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ***********************************************************
% Calculation of live tree removals
% ***********************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%*********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%*********************************************************************
% - all forested land
```

```
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%*********************************************************************
% CALCULATION OF CARBON POOLS
%*********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
```

```
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%**********************************************************************
% CALCULATION of SAWMILLS input and output
%**********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
        if a == 1
            statout(a,34) = smcap1(a) + smcap0;
        end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%**********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%**********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
```

```
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
```

```
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
%*********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%*********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%*********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
```

```
        flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
        flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
        flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
        b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
        flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
        man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
        flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
        flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
        flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    end
    % manure lbs per head per day (crosschecking)
    % for poultry
    % (flowmat(:,310,6)./statout(:,6))/365 * 2.2
    % Energy content calculated based on Channiwala 2002
    % HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
    % HHV in MJ/kg of dry mass
    % Estimation for H S O Ash is based on their relationship to carbon content
    % in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
    % and A/C = 0.6783
    % Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
    % check O/C = 0.95 approximately
    UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
    Mener   = zeros(3,1); % [cattle poultry swine]
    for a = 0:5:10
        b = a/5;
        Car = parause2(111+a);
        UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
        UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

        % HHV in MJ/kg of dry mass
        Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
            + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
            - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
        % Energy in kWh/y
        flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
        b = a/5;
        Car = parause2(111+a);
        UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
        UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

        % HHV in MJ/kg of dry mass
        Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
            + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
            - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
        % Energy in kWh/y
        flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
        b = a/5;
        Car = parause2(111+a);
        UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
        UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

        % HHV in MJ/kg of dry mass
        Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
            + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
            - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
        % Energy in kWh/y
        flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    end
    % TOTAL Manure Generated BEFORE handling and storage losses
    % [cattle poultry swine]
```

```
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%*********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%*********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
```

```
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
```

```
        rec_fert(a,b) = parause2(bB + b)./10;
     end
     bB = bB + b + 1;
     for b = 1:3 % [N P C]
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
        rec_fert(a,b) = parause2(bB + b)./10;
     end
     bB = bB + b + 1;
 end
 %*****************************************************************
 % Nutrient Content for each type of fertilizer IF SS PL CM SM CP
 %*****************************************************************
 % First we calculate the water content (moisture) of manure after storage
 % & handling (estimated in manure.m) as a fraction w/w of total mass,
 % nutrient content as w/w fraction of DM, and energy in kWh/tonne
 fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

 % Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
 % Column of zeros for IF
 for a = 1:nmat
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
     fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
         flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
 end
 % Now data is corrected to reflect fraction of mass. For compost we use the
 % typical compost composition entered by user
 % Dry matter in tonne/y
 fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
 % We convert first column from Water content Mgal/d to w/w of total mass
 fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

 % Nutrient content corrected to w/w of dry matter
 for a = 2:4
     fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
     fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
     fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
 end
 % Energy content corrected to kWh per tonne (fresh matter)
 fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

 % fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
 % fraction as from 1 to 5 are.
 % Data is added for IF (not calculated above) from user input
 % Assumed moisture 0.5%
 fm_cont(isnan(fm_cont)) = 0;
 fm_cont(:,1,1) = ones(x,1) * 0.005;
 fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

 % Data of ideal compost is added from user input
```

```
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
       + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
       - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated
```

```
% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
```

```
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);
```

```
%*********************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%*********************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*************************************************************
% MANURE distribution
%*************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
```

```
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%***************************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%***************************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%************************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);
```

```
% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%**************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%**************************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%**************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%**************************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%**************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%**************************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
```

```
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
```

```
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%**********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
```

```
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;
```

```
%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%**************************************************************************
% CONVERSIONS
%**************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
```

```
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%**********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
```

```
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
```

```
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
```

```
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  ************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  ************************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
```

```
        P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
        P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    end
    % Total meat produced
    flowmat(:,160,:) = sum(P_meat,2);

    % Total animal (meat) mass accumulated due to increase or decrease of
    % livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
    L_acc = zeros(x,3,nmat);
    % Average livestock growth in tonne/y, based on growth of thousand
    % heads: 'liveg' calculated in 'livepop.m'
    for b = 0:2
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
        L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    end
    % [year] x [Cattle Poultry Pigs] x [specie]
    for b = 0:4:8
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
        a = (b/4)+1;
        L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
        L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
        L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
        L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
        L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
        L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    end
    % Total livestock accumulation
    flowmat(:,306,:) = sum(L_acc,2);

    % DAIRY  ***********************************************************
    % Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
    % Food Materials (1896) Professor W.O. Atwater). For the total mass
    % will add again the refure mass.
    re_mass = 0.105;
    egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

    % Total dairy mass, milk and eggs in tonne per year
    mil_mass = statout(:,5).* statin(42)* statin(44);
    egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

    d_mass = [egg_mass mil_mass];
    % Matrix for diary production
    P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

    % For correlation between carbohydrates, protein, fat, nitrogen, carbon,
    % see notes page 53. Correlations derived from molecular formulas.
```

```
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);            % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d
```

```
% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);           % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ************************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  *************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
```

```
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
```

```
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*********************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*********************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
```

```
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
```

```
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%*********************************************************************
% FOOD SLUDGE FLOW
%*********************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%*********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%*********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
```

```
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
   flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
   flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
   flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%*********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%*********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);
```

```
%************************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%************************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%************************************************************************
% First part of the energy module. second part is 'energys2.m'
%************************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
```

```
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%*********************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%*********************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
```

```
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%*********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
```

```
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%**********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%**********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%**********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%**********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
```

```
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                           % content in raw coal (dry basis)
                           % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%**************************************************************************
% Emissions from power generation (coal)
%**************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%**************************************************************************
% Algae growing from coal plant emissions
%**************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
```

```
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%************************************************************************
% Production of biodiesel
%************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
```

```
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%**************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%**************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
```

```
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%*********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%*********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%*********************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%*********************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%*********************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%*********************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
      m_H_pl = 0;
      m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
```

```
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);
```

```
% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
```

```
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

% Checking: sum(pyFlows(:,1:8,1),2)
```

```
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                 % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                 % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                 % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                    % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                    % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                    % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%**********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%**********************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%**********************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%**********************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%**********************************************************************
```

```
% LANDFILLING
%*********************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
```

```
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%*********************************************************************
```

```
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%*********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%*********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%*********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
```

```
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
        % Emission to the atmosphere
        flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
        % Leaching to hydrosphere
        flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
        % Resulting composted fertilizer
        flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    end
    % Resulting composted fertilizer - CORRECTING TOTAL MASS
    % Substracting water and nutrient loss
    flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
    % check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
    % Energy content calculated based on Channiwala 2002
    % HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
    % HHV in MJ/kg of dry mass
    % Estimation for H S O Ash is based on their relationship to carbon content
    % in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
    % and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
    % compared to fresh manure ~20-30%. So we double of Singh's values,
    % Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
    % with the ~51% reduction in Carbon content during composting that promotes
    % a larger Ash/C ratio.
    % Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
    % check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
    UAn     = zeros(x,6);    % percentage [C H S O N Ash]
    Drym    = flowmat(:,315,6) - flowmat(:,315,7);
    UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
    UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
    UAn(:,5) = flowmat(:,315,2)./Drym * 100;
    UAn(:,6) = UAn(:,1) * 1.4;
    UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

    %Energy in MJ/kg
    comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
           + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
           - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

    % Energy in kWh/y
    flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;
```

```
% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%****************************************************************
% EXPORTS/IMPORTS
%****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
```

```
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%**********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%**********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
```

```
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%*********************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%*********************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
```

```
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
   flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
   flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
   tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
```

```
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
```

```
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%*********************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
```

```
%************************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%************************************************************************
% Water
%************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%************************************************************************
% Nitrogen
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
```

```
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
```

```
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%************************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%************************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%************************************************************************
% Forestry Sector
%************************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
```

```
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%*********************************************************************
% Food Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%*********************************************************************
% Energy Sector
%*********************************************************************
```

```
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%**********************************************************************
% Waste Management Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);
```

```
%************************************************************************
% INDICATOR SCORES ARE CALCULATED                                       *
%************************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%************************************************************************
%% NEW indicator
%************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
```

```
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
```

```
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
```

```
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim

caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
```

```
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%*************************************************************************
% Store data
%*************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%*************************************************************************
% Estimations associated with land uses
%*************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
```

```
%*********************************************************************
% ESTIMATION OF AREA for each designated land use
%*********************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
```

```
sstwuse =  4 * sst_flush;
%-------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
```

```
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%************************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%************************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
```

```
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% ************************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ************************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
       flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
           - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% ************************************************************************
% CALCULATION OF food consumption (by humans)
% ************************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%************************************************************************
% FOOD CONSUMPTION
%************************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
```

```
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';
```

```
% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
```

```
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
```

```
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%**************************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%**************************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);

% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
```

```
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%*********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%*********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
% Non-consumptive water use is discharged back to a surface source
%*********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%*********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
```

```
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
    % Consumptive use
    flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
    consutot = consutot + flowmat(:,a+49,1);
    % Power generation capacity time series
        for b = 1:x
            if b == 1
                statout(b,a+13) = inout(b,5+a) + statin(a+14);
            end
        end
    % Withdrawals
    flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
    withtot = withtot + flowmat(:,a+42,1);
```

```
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
end
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%*********************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
%*********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
```

```
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
```

```
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                    % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);              % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%*********************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%*********************************************************************
%
```

```
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
```

```
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            if Q_Et(luse_et,4) > precip
                Q_Et(luse_et,4) = precip;
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2

% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
```

```
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;
```

```
% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
end
% (b) sewer infiltration index
sinfil = parause(55) * sinfil;

% Runoff from pervious areas
flowmat(:,77,:) = epa - ginfil;

% Calculation of the fertilization mix (area-wise)
fertmix

tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
tpa = statout(:,27); % total pervious area km2
%tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
% Forested areas accounts also for urban forests
%*********************************************************************
% FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
%*********************************************************************
% Creates a matrix of the proportion of non-fertilized area for each land
% use with the size of the alternatives of fertilization
fractnF = zeros(tfe,size(Areas,2));
% [grass forestry crops]
for a = 1:tfe;
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
end

% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = a_mix.* (ones(tfe,3) - fractnF);
% fe_mix modified. non-fertilized area fraction is added.
```

```
% fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = cat(1,fractnF(1,:),fe_mix);
% depth = [number of years]
fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
for a = 1:x % each year
    for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%**********************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%**********************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y

% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
```

```
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%************************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%************************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%************************************************************************
% WATER OUTPUT CALCULATIONS
%************************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%************************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
```

```
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%*********************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
```

```
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%*********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%*********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap
```

```
% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
```

```
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%*********************************************************************
% Algae growing from WWTP effluent
%*********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
```

```
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%*********************************************************************
% Emissions to air
%*********************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;
```

```
%************************************************************************
% Activated sludge reactor
%************************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%************************************************************************
% Solid waste - SEWAGE SLUDGE GENERATION
%************************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ====================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
```

```
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%**************************************************************************
% BIOGAS FLOW
%**************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a

%**************************************************************************
% TREATED SLUDGE FLOW
%**************************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
```

```
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%**********************************************************************
% ENERGY USAGE BY WWTP
%**********************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*******************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
```

```
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*****************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    if edialog == 1;
    else
        emat(8) = 1;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
```

```
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ************************************************************
% Calculation of live tree removals
% ************************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
```

```
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%***********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%***********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%***********************************************************************
% CALCULATION OF CARBON POOLS
%***********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y

% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
```

```
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;

% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
        if a == 1
            statout(a,34) = smcap1(a) + smcap0;
```

```
        end
    end
    % Estimation of the maximum input capacity of sawmills based on the throuput
    % and the amount of byproducts (tonne/y). The flow of wood to sawmills is
    % assumed to be equal to their capacity.
    % input capacity includes wood and bark
    flowmat(:,101,6) = statout(:,34)./parause(74);
    % Nutrient and Energy content based on amount of bark and wood
    flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
        (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
    flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
    flowmat(:,101,1) = flowmat(:,101,7) /1382000;

    %*********************************************************************
    % CALCULATION OF SAWMILL OUTPUTS
    %*********************************************************************
    % Final product from sawmills to "Local Consumption" (tonne/y)
    flowmat(:,103,6) = statout(:,34);
    % Nutrient and Energy content based on wood composition only (no bark)
    flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
    flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
    flowmat(:,103,1) = flowmat(:,103,7) /1382000;

    %*********************************************************************
    % CALCULATION OF SAWMILL RESIDUE
    %*********************************************************************
    % Residue of sawmills
    flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
    %flowmat(:,101,2)./flowmat(:,101,6)
    %flowmat(:,103,2)./flowmat(:,103,6)
    % Sawmill residue not used, i.e. exported (negative value)
    flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
    % Sawmill residue used as industrial fuel
    flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
    % Sawmill residue used as a product in the local market
    flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
    %flowmat(:,102,2)./flowmat(:,102,6)
    %*********************************************************************
    % CALCULATION OF WOOD for PAPER PRODUCTION
    %*********************************************************************
    % Paper production and composition
    flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
    % Nutrient and Energy content based on paper composition
    flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
    flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
    flowmat(:,327,1) = flowmat(:,327,7) /1382000;

    % Input of wood for paper production - a function of the pulp extraction
    % process as described by parause(73)
    flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
    % Nutrient and Energy content based on wood composition only (no bark)
    flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
    flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
    flowmat(:,328,1) = flowmat(:,328,7) /1382000;

    %*********************************************************************
    % CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
    %*********************************************************************
    % Estimation of the annual internal use of wood products based on
    % population (in metric tonnes).
    flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
```

```
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%*********************************************************************
% ATMOSPHERIC PROCESSES
%*********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo
```

```
nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%*********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%*********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
```

```
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%**********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%**********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
```

```
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);
```

```
%*****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%*****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
end
%*****************************************************************
% Nutrient Content for each type of fertilizer IF SS PL CM SM CP
%*****************************************************************
% First we calculate the water content (moisture) of manure after storage
% & handling (estimated in manure.m) as a fraction w/w of total mass,
% nutrient content as w/w fraction of DM, and energy in kWh/tonne
fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

% Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
% Column of zeros for IF
for a = 1:nmat
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
end
% Now data is corrected to reflect fraction of mass. For compost we use the
% typical compost composition entered by user
% Dry matter in tonne/y
fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
% We convert first column from Water content Mgal/d to w/w of total mass
```

```
fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

% Nutrient content corrected to w/w of dry matter
for a = 2:4
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
end
% Energy content corrected to kWh per tonne (fresh matter)
fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

% fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
% fraction as from 1 to 5 are.
% Data is added for IF (not calculated above) from user input
% Assumed moisture 0.5%
fm_cont(isnan(fm_cont)) = 0;
fm_cont(:,1,1) = ones(x,1) * 0.005;
fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

% Data of ideal compost is added from user input
fm_cont(:,6,1) = parause2(128)/100;
fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
% energy calculated from ultimate analysis
% Energy in kWh per tonne of total mass (wet basis)
EneComp % fm_cont(:,6,5) calculated

UAn     = zeros(x,6);    % percentage [C H S O N Ash]

UAn(:,1) = fm_cont(:,6,4) * 100;
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = fm_cont(:,6,2) * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
        + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
        - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh per tonne of total mass (wet basis)
fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

%*****************************************************************
% NUTRIENT APPLICATION RATE
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];
```

```
% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
```

```
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
```

```
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%**************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%**************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
```

```
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%*********************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%*********************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
```

```
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%************************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%************************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%************************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%************************************************************************
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
```

```
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;
```

```
% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%**********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;
```

```
% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%**************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
```

```
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%************************************************************************
% CONVERSIONS
%************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
```

```
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%**********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%**********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
```

```
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
```

```
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
```

```
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*********************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
%*********************************************************************
% FISH  ************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  ************************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
```

```
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
end
% Total meat produced
flowmat(:,160,:) = sum(P_meat,2);

% Total animal (meat) mass accumulated due to increase or decrease of
% livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
L_acc = zeros(x,3,nmat);
% Average livestock growth in tonne/y, based on growth of thousand
% heads: 'liveg' calculated in 'livepop.m'
for b = 0:2
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
end
% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
end
% Total livestock accumulation
```

```
flowmat(:,306,:) = sum(L_acc,2);

% DAIRY  ************************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);

% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
```

```
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);           % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);           % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d

% VEGGIES  ***********************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
```

```
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  ************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% ------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
```

```
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%-------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%*********************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%*********************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
```

```
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];
```

```
% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%************************************************************************
% BIOGAS FLOW
%************************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%************************************************************************
% FOOD SLUDGE FLOW
%************************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
```

```
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%**********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%**********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
```

```
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%*********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%*********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%*********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%*********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%*********************************************************************
% First part of the energy module. second part is 'energys2.m'
%*********************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
```

```
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));

%************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
```

```
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%*********************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%*********************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%*********************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
```

```
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%**********************************************************************
% Estimation of flows for power generation
%**********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%**********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%**********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix

% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
```

```
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%*********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%*********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                            % content in raw coal (dry basis)
                            % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);
```

```
% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%************************************************************************
% Algae growing from coal plant emissions
%************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;

%************************************************************************
% Production of biodiesel
%************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
```

```
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%**************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%**************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);

% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
```

```
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%**********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%**********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**********************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%**********************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
```

```
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%************************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%************************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
```

```
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
```

```
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
   pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
   pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
   pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
       + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
       - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
   pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
   pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
   pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
       pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
```

```
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

% Checking: sum(pyFlows(:,1:8,1),2)
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                   % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                   % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                   % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;
```

```
% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%**********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%**********************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%**********************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%**********************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%**********************************************************************
% LANDFILLING
%**********************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];
```

```
% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;

% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
```

```
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%**************************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%**************************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%**************************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%**************************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);

% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
```

```
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
end
% Resulting composted fertilizer - CORRECTING TOTAL MASS
% Substracting water and nutrient loss
flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
% check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
% and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
% compared to fresh manure ~20-30%. So we double of Singh's values,
```

```
% Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
% with the ~51% reduction in Carbon content during composting that promotes
% a larger Ash/C ratio.
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
UAn     = zeros(x,6);    % percentage [C H S O N Ash]
Drym    = flowmat(:,315,6) - flowmat(:,315,7);
UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = flowmat(:,315,2)./Drym * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
       + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
       - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%****************************************************************
% EXPORTS/IMPORTS
%****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%****************************************************************
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);
```

```
%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%*********************************************************************
```

```
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
    else
        if impDI(a) < 0 && isnan(impDI(a))
            end
         else % impDI(a) > 0 or diesel has been used up for power
            % all diesel (fossil and bio) is imported
            flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported diesel (fossil)
            flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
            % imported diesel (bio)
            flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
            flowmat(a,247,:) = flowmat(a,241,:);
        end
    end
end
% Gasoline characteristics and import/export estimation
flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
```

```
% 240   Imports/exports of gasoline (fossil fuel only and no production)
% Assumes also that the minimum content of biofuels is achieved (imported)
flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
% Import of bio-gasoline (includes biofuel for diesel engines)
flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
% Total biofuel used for engines (gasoline and diesel)
flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

%*****************************************************************
% ESTIMATION OF EMISSIONS from local use and power
%*****************************************************************
% Emissions from fuel consumption in tonne/y (non power)
emissionF       % by FUEL TYPE

for a = 1:nmat
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
```

```
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE

for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
```

```
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;
```

```
flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%************************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%************************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%************************************************************************
% Water
%************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%************************************************************************
% Nitrogen
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);
```

```
%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;
```

```
% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%************************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%************************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
```

```
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%*********************************************************************
% Forestry Sector
%*********************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%*********************************************************************
% Food Sector
%*********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;

I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
```

```
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%**********************************************************************
% Energy Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%**********************************************************************
% Waste Management Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;
```

```
% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%*********************************************************************
% INDICATOR SCORES ARE CALCULATED                                    *
%*********************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%*********************************************************************
%% NEW indicator
%*********************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
```

```
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
```

```
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;

      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
```

```
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

      % Parameters with fixed value
      para_both(to_fix)    = paracat(to_fix,2);

      % Parameters sampled
      para_both(to_sample) = abso(sa,:);  % size(abso) size(to_sample)

      % We now divide both parameter groups: (1) model parameters, (2) flow
      % composition. We s eparate them based on the original row number,
      % 'pg1' and 'pg2' to generate two parameter vectors as needed by the
      % model (see rimmeb.m)

      parause   = para_both(pg1)';
      parause2  = para_both(pg2)';
      statin    = para_both(pg3)';
      inout     = reshape(para_both(pg4)',len,x); % 'len' is the length of inout
      inout     = inout';

      %MSA main algorithm (The model)
      msa_main

prelim
```

```
caltopo

p0 = statin(1);
% 'max population' assumes that population doubles every 20 years
pmax = statin(1) + statin(1) * x/20;
time = (0:x-1);
kgrowth = parause(1);
topo = zeros(1,x);

% Population growth is estimayed based on the model selected by the user
% via  modset(12)
    if modset(12) == 1
    end

    if modset(12) == 2
    % Constant percentage model
    topo(1) = p0;
    for a = 2:x
    end
    end



    if modset(12) == 3
    end

topo = topo';
%
%************************************************************************
% Store data
%************************************************************************
% Output file stores in vertical form
% years in simulation
statout(:,1) = time'+  modset(2);
% total population
statout(:,2) = topo;
% Self-supplied Population (water)
statout(:,3) = topo * parause(3);
% Public-supplied Population (water)
statout(:,4) = topo - statout(:,3);

%************************************************************************
% Estimations associated with land uses
%************************************************************************
landuse

land = statin(2);

% Matrix of land fractions
% Forested Areas                1
% low intensity urban areas     2
% high intensity urban areas    3
% crop and pastures             4
% open water and wetland areas  5
% clear cut and sparse areas    6
land_fract = zeros(x+1,6);

% Fractions first year (begining of year)
land_fract(1,1:5) = statin(9:13);
```

```
land_fract(1,6) = 1 - sum(land_fract(1,1:5));

%Checking that the sum of fractions is less than 1
if sum(land_fract(1,1:5)) > 1
    end
else
    emat(1) = 0;
end
% Land fractions variating in time
% inout(19 inout(20 inout(21 inout(22 inout(23 are % of annual change
if x > 1
end
end

% Checking that the resulting 'land use 6' is positive for all years
if land_fract(:,6) < 0
    end
else
    emat(2) = 0;
end
%*********************************************************************
% ESTIMATION OF AREA for each designated land use
%*********************************************************************
% statout(:,8 statout(:,9 statout(:,10 statout(:,11 statout(:,12
% statout(:,13
statout(:,8:13)  = land * land_fract(1:x,:);

% Impervious Areas (correlated to the percentage of low intensity and high
% intensity urban areas). Km2
% For the UC Watershed the correlation coefficient is 0.99972 using an
% exponential relationship
% Percentage
toturban = (land_fract(1:x,2) + land_fract(1:x,3)) * 100;
% Fraction for UCW
% fimpervious = 0.01520 * exp(toturban.* 0.06234);
% Data from user and corrected for each year with the urban area
% This assumes that the impervious area density does not change over time
fimpervious = statin(4) * toturban / toturban(1);
statout(:,26) = land * fimpervious/100;

% Pervious Areas: Estimated as: total area -open water -impervious area
statout(:,27) = land - statout(:,12) - statout(:,26);

%*********************************************************************
% ESTIMATION OF TIMBERLAND LAND AREA (as a function of forest areas)
%*********************************************************************
statout(:,31) = statout(:,8) * statin(35);

% MODULE Water sector
waters;

sspo = statout(:,2) * parause(3);
%Public supplied population
pspo = statout(:,2) * (1 - parause(3));

% Urine Separation Calculations
ur_sep

flowmat(:,127,6) = (statout(:,2) * 1000) * parause(95) * 365 / 1000;
% Total solids in urine (tonne/year)
TSurine = flowmat(:,127,6) * (1 - parause(96));
```

```
% Constituents in urine (W, N, P, C, E) Mgal/d and tonne/y
flowmat(:,127,1) = flowmat(:,127,6) * parause(96) / 1382000 ;
flowmat(:,127,7) = flowmat(:,127,1) * 1382000 ;
for a = 1:x
    flowmat(a,127,2:4) = TSurine(a).* parause2(5:7)./ 100;
    % Energy calculated base on Mary 1972, 10.12 kcal/g N
    flowmat(a,127,5) = flowmat(a,127,2).* 11760;
end
%*********************************************************************
% Calculation of URINE SEPARATION TECHNOLOGY flows
%*********************************************************************
% Urine-related flushes = 4 per person per day. Based on Lienert 2006 (in
% Env Sci Tech, issue 40), that estimates 80 L per family of four per day,
% then we assumed a modern water efficient toilet of 6 L per flush.
nst_flush = 6.00 / 3.7854117834; % Water usage of non-Separation Toilet in gal per flush
sst_flush = 0.15 / 3.7854117834; % Water usage of sst in gal per flush
% Water use if SST is NOT implemented, gal per cap/d
nstwuse = 4 * nst_flush;
% Water use if SST implemented, gal per cap/d
sstwuse =  4 * sst_flush;
%---------------------------------------------------------------
% Mgal/d, how much water is used and diverted to urine-fertilizer facility
sstwuse_a = parause(98) * (statout(:,2) * 1000) * sstwuse  /1000000;

% (Urine + flushed water) flow diverted by SST regardless people are
% conected to a septic system or to the sewer network.
flowmat(:,131,1)    = sstwuse_a + flowmat(:,127,1) * parause(97) * parause(98); % Mgal/d
flowmat(:,131,2:5)  = flowmat(:,127,2:5) * parause(97) * parause(98); % tonne/y
flowmat(:,131,6)    = flowmat(:,131,1)* 1382000;  % tonne/y
flowmat(:,131,7)    = flowmat(:,131,6); % Assumed as water

% The Struvite process
struProd

EneStru = parause(218) * 3785.3846; % kWh/MG
% Energy requirement based on total flow from urine separating toilets
%flowmat(:,334,5) = flowmat(:,131,1).* (365 * EneStru); % kWh/a
% Energy requirement based on only urine flow (no flushing water)
flowmat(:,334,5) = flowmat(:,127,1) *parause(97) *parause(98)* (365 * EneStru); % kWh/a

% tonne Moles of N, P, C, and Water and accounting for the struvite process
% efficiency with respect N and P
stru_moles = [flowmat(:,131,2) * parause(222) ...
    flowmat(:,131,3)* parause(223) flowmat(:,131,4)...
    flowmat(:,131,7)]./ (ones(x,1) * [14 31 12 18]);

% Estimation of struvite production
struvite = zeros(x,1,nmat);
AmmSulph = zeros(x,1,nmat);
WaterStru = zeros(x,1,nmat);

% Formation of Mg(NH4)PO4.6H2O (struvite), MW = 827.31
% Limiting nutrient is one with the lowest number (between N and P)
[val3,NutInt3] = min(stru_moles(:,1:2),[ ],2);
struvite(:,1,2:7) = diag(stru_moles(:,NutInt3)) * [14 31 0 0 827.31 6*18];
struvite(:,1,1) = struvite(:,1,7) / 1382000;

% Formation of (NH4)2SO4 (Ammonium Sulphate), MW = 132.14
% Using the N left from struvite formation
%if NutInt3 % P is limiting
```

```
    AmmSulph(NutInt3 > 1,1,2) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 14;
    AmmSulph(NutInt3 > 1,1,6) = (stru_moles(NutInt3 > 1,1) - stru_moles(NutInt3 > 1,2)) * 132.14;
%AmmSulph(NutInt3 > 1,1,6) = (stru_moles(:,1) - stru_moles(:,2)) * 132.14;
%else % N is limiting, is was used up for struvite
    AmmSulph(NutInt3 < 2,1,:) = 0; % N is limiting, is was used up for struvite
%end
% Emissions from struvite process
% complete. CO2 stripping emissions
flowmat(:,341,4) = flowmat(:,131,4) * parause(224);

% Material left after the Struvite process
stru_relea = flowmat(:,131,:) - struvite - AmmSulph - flowmat(:,341,:);

% Modified to include changes in consumption over time
chdow = ones(x,1) * parause(2);
if x > 1
    end
end
% DO PS drinking water use
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    % tonne/y
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382;
end
% Self-supplied (SS) consumption is the same as public supplied
% DO Consumption SS population (from ground water sources)
flowmat(:,10,1) = (sspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,10,2:4) = flowmat(a,10,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total DO water use
flowmat(:,11,:) = flowmat(:,7,:) + flowmat(:,10,:);

% Self supplied withdrawals for Domestic use GW
flowmat(:,8,:) = flowmat(:,10,:)* parause(4);
% Self supplied withdrawals for Domestic use SW
flowmat(:,9,:) = flowmat(:,10,:) - flowmat(:,8,:);

% ************************************************************************
% DO consumptive water use - *** SEPTIC TANKS ***
% ************************************************************************
% Population using septic tanks 'sep_pop'
% Estimates the population using septic tanks
% Equation is derived from the JJ&G equation
% proposed for the metropolitan GA water
% planning distric (year 2000)
% density in capita per acre
density = topo.*(1000 / (land * 247.105381));

% Fraction of population using septic tanks
fractpop = 0.87425 * exp(density.*-0.8256);

sep_pop = topo.* fractpop;

% Water consumption for septic tank users
flowmat(:,12,1)   = sep_pop.* chdow / 1000; % Mgal/d

% Calculates the load of N, P, C in tonne/y BEFORE WATER IS USED
% concentration from the water source
aveconc = parause2(89:91).* parause(3) + parause2(81:83).*(1 - parause(3));
for a = 1:x
```

```
    flowmat(a,12,2:4) = flowmat(a,12,1) * aveconc * 1.382; % tonne/y
end
%*********************************************************************
% SEWAGE AND MUNICIPAL SOLID WASTE
%*********************************************************************
% Calculates the load of N, P, C in tonne/y AFTER WATER IS USED
% concentration of source + wastes in water
do_waste

flowmat(:,128,6) = (statout(:,2) * 1000) * parause(99) * 365 / 1000;
% Total solids in faeces (tonne/year)
TSfaeces = flowmat(:,128,6) * (1 - parause(100));
% Constituents in faeces (W, N, P, C, E) - Total amount possible
% Mgal/d of water
flowmat(:,128,1) = flowmat(:,128,6) * parause(100) / 1382000 ;
% tonne/y of substance
for a = 1:x
    flowmat(a,128,2:4) = TSfaeces(a) * parause2(9:11)./ 100;
end
% Energy content is based on wet weight
flowmat(:,128,5) = flowmat(:,128,6) * parause2(12);

% ********************************************************************
% CALCULATION FOR POPULATION USING SEPTIC TANKS
% ********************************************************************
% Amount of nutrients sent to septic tanks (tonne/y):
% (faeces + urine - urine separated SST)ALL POPULATION + DRINKING WATER NUTRIENT LOAD.
% Note: this assumes that source separation (SST) implementation is same
% for both groups: septic-tank-users and non-septic-tank-users
for a = 1:x
        flowmat(a,12,2:5) = (flowmat(a,128,2:5) + flowmat(a,127,2:5)...
            - flowmat(a,131,2:5)).* (sep_pop(a)./statout(a,2)) + flowmat(a,12,2:5);
end
flowmat(:,12,6) = flowmat(:,12,1) * 1382000; % tonne/y (approx to water density)
flowmat(:,318,2:4) = flowmat(:,12,2:4).* -parause(115);
flowmat(:,318,3) = flowmat(:,318,3).* 0;

% ********************************************************************
% CALCULATION OF food consumption (by humans)
% ********************************************************************
food1

m_meat = [0.73 0.64 0.55]; % Water content [cattle poultry pigs]
% Water content in dairy
m_dai = [0.735 0.87];  % [eggs milk]
% Water content in food varies enourmously. Fruit can vary from 75%
% (banana) to 92% (strawberry). Veggies can varies from 80% to 95%.
% Cereals, on the other hand can contain between 13% and 15% of moisture.
% Fresh fish has a moisture of 65-79%. 'Others' is assumed to have 95%.
% [fruits cereal veggies B-meat Po-meat Pi-meat fish eggs milk others]
m_food = [fr_mix(1)/100 ce_mix(1)/100 ve_mix(1)/100 m_meat(1) m_meat(2)...
    m_meat(3) 0.72 m_dai(1) m_dai(2) 0.95];

%*********************************************************************
% FOOD CONSUMPTION
%*********************************************************************
% Estimates food consumption based on population. Mass of food consumption
% (as reported by FAO) are assumed as bulk weight (dry matter + moisture)
% and compositions are associated to this bulk weight.
%
```

```
% [fruits cereal veggies B-meat Po-meat Pi-meat fish dairy others]
% Allocating Matrix space
C_meat = zeros(x,3,nmat);  % [years] x [cattle poultry pigs] x [specie]
C_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]
% Based on FAO data for US, 5% of the mass of dairy consumed corresponds to
% egg, while 95% is milk and butter. Cheese is part of milk consumption.
egda = 0.05;

% Mass calculation all in tonnes/y
% flowmat(:,149 flowmat(:,150 flowmat(:,151 flowmat(:,152 flowmat(:,153
for a = 1:x
    flowmat(a,149:151,6) = parause(149:151).* statout(a,2);
    % Detailed meat calculation
    C_meat(a,:,6) = parause(152:154).* statout(a,2);
    % Meat is compounded into one term
    flowmat(a,152,6) = sum(C_meat(a,:,6));
    % Rest of food items
    flowmat(a,153:155,6) = parause(155:157).* statout(a,2);
    % Detailed dairy calculation
    C_dairy(a,:,6) = [flowmat(a,153,6)*egda flowmat(a,153,6)*(1 - egda)];
end
% Composition of each food group
for a = 1:x
% [fruits cereal veggies]
    flowmat(a,149:151,2) = flowmat(a,149:151,6).* parause2(149:4:157)'; % tonnes/y
    flowmat(a,149:151,3) = flowmat(a,149:151,6).* parause2(150:4:158)'; % tonnes/y
    flowmat(a,149:151,4) = flowmat(a,149:151,6).* parause2(151:4:159)'; % tonnes/y
    flowmat(a,149:151,5) = flowmat(a,149:151,6).* (parause2(152:4:160)*1.1622232)'; % kWh/y
    flowmat(a,149:151,7) = flowmat(a,149:151,6).* m_food(1:3); % tonnes/y
    flowmat(a,149:151,1) = flowmat(a,149:151,7)./ 1382000; % Mgal/d
end
for a = 1:x
% [B-meat Po-meat Pi-meat]
% composition used is same as live animal (wet weight)
    C_meat(a,:,2) = C_meat(a,:,6).* parause2(189:4:197)'; % tonnes/y
    C_meat(a,:,3) = C_meat(a,:,6).* parause2(190:4:198)'; % tonnes/y
    C_meat(a,:,4) = C_meat(a,:,6).* parause2(191:4:199)'; % tonnes/y
    C_meat(a,:,5) = C_meat(a,:,6).* parause2(192:4:200)' * 1.1622232; % kWh/y
    C_meat(a,:,7) = C_meat(a,:,6).* m_food(4:6); % tonnes/y
    C_meat(a,:,1) = C_meat(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,152,:) = sum(C_meat(:,:,:),2);
end
for a = 1:x
% [eggs milk]
% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
    C_dairy(a,:,2) = C_dairy(a,:,6).* (parause2(216:5:221)./100)'/6.25;  % tonnes/y
    C_dairy(a,:,3) = C_dairy(a,:,6).* (parause2(219:5:224)./100)';  % tonnes/y
    C_dairy(a,:,4) = C_dairy(a,:,6).* ((parause2(216:5:221)'/100)/1.98 +
(parause2(217:5:222)'/100)/1.357 + (parause2(218:5:223)'/100)/2.50);  % tonnes/y
    C_dairy(a,:,5) = C_dairy(a,:,6).* parause2(220:5:225)' * 1.1622232;  % kWh/y
    C_dairy(a,:,7) = C_dairy(a,:,6).* m_food(8:9); % tonnes/y
    C_dairy(a,:,1) = C_dairy(a,:,7)./ 1382000; % Mgal/d
    flowmat(:,154,:) = sum(C_dairy(:,:,:),2);
end
for a = 1:x
% [fish other]
    flowmat(a,153:2:155,2) = flowmat(a,153:2:155,6).* parause2(161:4:165)'; % tonnes/y
    flowmat(a,153:2:155,3) = flowmat(a,153:2:155,6).* parause2(162:4:166)'; % tonnes/y
    flowmat(a,153:2:155,4) = flowmat(a,153:2:155,6).* parause2(163:4:167)'; % tonnes/y
    flowmat(a,153:2:155,5) = flowmat(a,153:2:155,6).* (parause2(164:4:168)* 1.1622232)'; % kWh/y
```

```
    flowmat(a,153:2:155,7) = flowmat(a,153:2:155,6).* m_food(7:3:10); % tonnes/y
    flowmat(a,153:2:155,1) = flowmat(a,153:2:155,7)./ 1382000; % Mgal/d
end
% Total food for human consumption (consumed)
flowmat(:,156,:) = sum(flowmat(:,149:155,:),2);

% ************************************************************************
% CALCULATION OF MUNICIPAL WASTE
% ************************************************************************
mswcalc1

flowmat(:,107,6) = statout(:,2) * parause(84) * 365;

% paper disposed (moisture generally 3-7%)
flowmat(:,108,6)   = flowmat(:,107,6) * parause(85);
flowmat(:,108,7)   = flowmat(:,108,6) * parause2(148);
flowmat(:,108,1)   = flowmat(:,108,7) / 1382000;        % Mgal/d
% Composition data in dry basis
flowmat(:,108,2:5) = (flowmat(:,108,6) - flowmat(:,108,7)) * parause2(144:147)';

% ------------------------------------------------------------------------
% food disposed
flowmat(:,109,6) = flowmat(:,107,6) * parause(86);
% For MSW food we assume a composition accordingly to the intake of food
food_comp = zeros(1,nmat);
for a = 1:nmat
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
    food_comp(a) = flowmat(1,156,a)./flowmat(1,156,6);
end
flowmat(:,109,:) = flowmat(:,109,6) * food_comp;
% Correction for amount of organic waste sent to the sewer system
% Determined by parause(67)
flowmat(:,330,:) = flowmat(:,109,:) * parause(67);
flowmat(:,109,:) = flowmat(:,109,:) * (1 - parause(67));
% ------------------------------------------------------------------------
% wood disposed ('dry' lumber moisture generally less than 19%)
flowmat(:,110,6)    = flowmat(:,107,6) * parause(87);
flowmat(:,110,7)    = flowmat(:,110,6) * parause2(133);
flowmat(:,110,1)    = flowmat(:,110,7) / 1382000;
% Composition data in dry basis
flowmat(:,110,2:5)  = (flowmat(:,110,6) - flowmat(:,110,7)) * parause2(129:132)';

% yard waste disposed
flowmat(:,111,6) = flowmat(:,107,6) * parause(88);
flowmat(:,111,7)   = flowmat(:,111,6) * parause2(143);
flowmat(:,111,1)   = flowmat(:,111,7) / 1382000;
flowmat(:,111,2:4) = (flowmat(:,111,6) - flowmat(:,111,7)) * parause2(139:141)';
flowmat(:,111,5)   = (flowmat(:,111,6) - flowmat(:,111,7)).* parause2(142);

% DO consumptive water use - GARDEN & LAWN
flowmat(:,13,:) = flowmat(:,11,:) * parause(5);
% DO Total consumptive water use
flowmat(:,14,:) = flowmat(:,12,:) + flowmat(:,13,:);

%*********************************************************************
```

```
% COMMERCIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total CO water use
% Modified to include changes in consumption over time
chcow = ones(x,1) * parause(6);
if x > 1
    end
end
% Total CO water use
flowmat(:,19,1) = statout(:,2).* chcow / 1000;
% Self supplied withdrawals for Commercial use GW + SW
flowmat(:,18,1) = flowmat(:,19,1)* parause(7);

% CO delivered from public supply PS
flowmat(:,15,1) = flowmat(:,19,1) - flowmat(:,18,1);
for a = 1:x
    flowmat(a,15,2:4) = flowmat(a,15,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use GW
flowmat(:,16,1) = flowmat(:,18,1) * parause(8);
for a = 1:x
    flowmat(a,16,2:4) = flowmat(a,16,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Commercial use SW
flowmat(:,17,1) = flowmat(:,18,1) - flowmat(:,16,1);
for a = 1:x
    flowmat(a,17,2:4) = flowmat(a,17,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied water CO use from GW (COMPOSITION)
flowmat(:,18,2:4) = flowmat(:,16,2:4) + flowmat(:,17,2:4);
% Total CO water use (COMPOSITION)
flowmat(:,19,2:4) = flowmat(:,15,2:4) + flowmat(:,16,2:4) + flowmat(:,17,2:4);

% Total CO consumptive water use (ASSUMED UNIFORM PARTITION FROM SOURCES)
flowmat(:,20,:) = flowmat(:,19,:) * parause(9);

%*********************************************************************
% PUBLIC WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Drinking water for public use and losses
% Modified to include changes in consumption over time
chpuw = ones(x,1) * parause(10);
if x > 1
    end
end
% Drinking water for public use and losses
flowmat(:,23,1) = statout(:,2).* chpuw / 1000;
for a = 1:x
    flowmat(a,23,2:4) = flowmat(a,23,1) * parause2(81:83) * 1.382; % tonne/y
end
% Drinking water losses from public use (consumptive)
flowmat(:,22,:) = flowmat(:,23,:) * parause(11);
% Drinking water for indoor public use
flowmat(:,21,:) = flowmat(:,23,:) - flowmat(:,22,:);

%*********************************************************************
% INDUSTRIAL WATER USE CALCULATIONS (population dependent)
%*********************************************************************
% Total Industrial water use
% Modified to include changes in consumption over time
chinw = ones(x,1) * parause(12);
```

```
if x > 1
    end
end
% Total Industrial water use (NOT power generation)
flowmat(:,28,1) = statout(:,2).* chinw / 1000;
% Self supplied withdrawals for Industrial use GW + SW
flowmat(:,27,1) = flowmat(:,28,1)* parause(14);

% PS drinking water for industrial use
flowmat(:,24,1) = flowmat(:,28,1) - flowmat(:,27,1);
for a = 1:x
    flowmat(a,24,2:4) = flowmat(a,24,1) * parause2(81:83) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use GW
flowmat(:,25,1) = flowmat(:,27,1) * parause(15);
for a = 1:x
    flowmat(a,25,2:4) = flowmat(a,25,1) * parause2(89:91) * 1.382; % tonne/y
end
% Self supplied withdrawals for Industrial use SW
flowmat(:,26,1) = flowmat(:,27,1) - flowmat(:,25,1);
for a = 1:x
    flowmat(a,26,2:4) = flowmat(a,26,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total Industrial water use (COMPOSITION)
flowmat(:,28,2:4) = flowmat(:,24,2:4) + flowmat(:,25,2:4) + flowmat(:,26,2:4);

% Direct IN discharge to water sources (Assumed that no nutrient is
% introduced by the industrial process)
flowmat(:,29,:) = flowmat(:,28,:) * parause(13);
% Direct IN consumptive use (evaporation, and no nutrient value)
flowmat(:,30,:) = -1 * flowmat(:,28,:) * parause(16);
for a = 2:4
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
    flowmat(:,30,a) = 0;
end

%*********************************************************************
% MINING AND LIVESTOCK WATER USE CALCULATIONS
%*********************************************************************
% Both uses are ASSUMED TO BE Self-Supplied
% Total water use Mining (MI): GW + SW
% Assuming a linear behavior
flowmat(:,33,1) = statout(:,1) * parause(17) + statin(5);

% Water withdrawals from GW
flowmat(:,31,1) = flowmat(:,33,1) * parause(18);
for a = 1:x
    flowmat(a,31,2:4) = flowmat(a,31,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,32,1) = flowmat(:,33,1) - flowmat(:,31,1);
for a = 1:x
    flowmat(a,32,2:4) = flowmat(a,32,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Mining (MI) - COMPOSITION
flowmat(:,33,2:4) = flowmat(:,31,2:4) + flowmat(:,32,2:4);
% MI consumtive use
flowmat(:,34,:) = flowmat(:,33,:) * parause(19);
```

```
% Total water use Livestock (LI): GW + SW
% Calculated as a function of livestock inventory and water consumption per
% animal based on its type (cattle, poultry, or pigs)
livepop

livestock = zeros(x + 1,3); % [years + 1] x [Cattle Poultry Pigs]
liveg = zeros(x,3); % [years] x [Cattle Poultry Pigs]

% livestock for begining of first year (thousands)
livestock(1,:) = [statin(6) statin(7) statin(8)];
% Livestock annual inventory growth factor
live_gf = [parause(25) parause(26) parause(27)]./100;

% statout(:,5 statout(:,6 statout(:,7)
for a = 1:x
    % Inventory heads (begining and end of year)
    livestock(a+1,:) = livestock(a,:) + livestock(a,:).* live_gf;
    % Average Inventory heads in thousands
    statout(a,5:7) = 0.5 * (livestock(a,:) + livestock(a+1,:));
    % Average livestock growth in thousand heads [year] x [C Po Pi]
    liveg(a,:) = livestock(a+1,:) - livestock(a,:);
end
%**********************************************************************
% ESTIMATION OF WATER CONSUMPTION FOR LIVESTOCK
%**********************************************************************
% Water consumption is estamated based on the average annual animal
% inventory for each livestock category
% Cattle
cat_W = statout(:,5).* parause(22).* 0.001;       % Mgal/d
% Poultry
pou_W = statout(:,6).* parause(23).* 0.001;       % Mgal/d
% Pigs
pig_W = statout(:,7).* parause(24).* 0.001;       % Mgal/d

% Total LI water use
flowmat(:,37,1) = (cat_W + pou_W + pig_W);

% Water withdrawals from GW
flowmat(:,35,1) = flowmat(:,37,1) * parause(20);
for a = 1:x
    flowmat(a,35,2:4) = flowmat(a,35,1) * parause2(89:91) * 1.382; % tonne/y
end
% Water withdrawals from SW
flowmat(:,36,1) = flowmat(:,37,1) - flowmat(:,35,1);
for a = 1:x
    flowmat(a,36,2:4) = flowmat(a,36,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water use Livestock (LI) - COMPOSITION
flowmat(:,37,2:4) = flowmat(:,35,2:4) + flowmat(:,36,2:4);
% MI consumtive use
flowmat(:,38,:) = flowmat(:,37,:) * parause(21);

% Total OTHERS MI + LI
% total Water use
flowmat(:,41,:) = flowmat(:,33,:) + flowmat(:,37,:);
% total water withdrawals from GW
flowmat(:,39,:) = flowmat(:,31,:) + flowmat(:,35,:);
% total water withdrawals from SW
flowmat(:,40,:) = flowmat(:,32,:) + flowmat(:,36,:);
% total consumtive use
flowmat(:,42,:) = flowmat(:,34,:) + flowmat(:,38,:);
```

```
% Non-consumptive water use is discharged back to a surface source
%*********************************************************************
% POWER GENERATION WATER USE CALCULATIONS (capacity dependent)
%*********************************************************************
% Hydropower is assumed to have no water use as no withdrawal is associated
%
% Water use associated to 7 types of power generation processes: coal, ng,
% nuclear, diesel, alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;
for a = 1:7
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
```

```
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
% Power generation capacity time series
    for b = 1:x
        if b == 1
            statout(b,a+13) = inout(b,5+a) + statin(a+14);
        end
    end
% Withdrawals
flowmat(:,a+42,1) = statout(:,a+13)* (24/1000)* parause(a+27);
withtot = withtot + flowmat(:,a+42,1);
% Consumptive use
flowmat(:,a+49,1) = statout(:,a+13)* (24/1000)* parause(a+34);
consutot = consutot + flowmat(:,a+49,1);
end
% Correction for air cooling technology for natural gas power generation
flowmat(:,44,1) = flowmat(:,44,1) * (1 - statin(52));

% Total water withdrawal GW + SW
flowmat(:,59,1) = withtot;
%Total water withdrawn GW
flowmat(:,57,1) = flowmat(:,59,1) * parause(42); % Mgal/d
for a = 1:x
    flowmat(a,57,2:4) = flowmat(a,57,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,58,1) = flowmat(:,59,1) - flowmat(:,57,1); % Mgal/d
for a = 1:x
    flowmat(a,58,2:4) = flowmat(a,58,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,59,2:4) = flowmat(:,57,2:4) + flowmat(:,58,2:4) ;

% Total consumptive water use
flowmat(:,60,1) = -1 * consutot;
% Total water discharged from power generation processes
flowmat(:,61,:) = flowmat(:,59,:) +  flowmat(:,60,:);

%*****************************************************************
% FUEL PRODUCTION WATER USE CALCULATIONS
```

```
%*********************************************************************
% Water use associated to 3 types of fuel production (hypothetical)
% processes: alt 1, alt 2, and alt 3
withtot = 0;
consutot = 0;

for a = 1:3
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
% Fuel production capacity time series
%statout(:,i+21) = inout(:,12+i) + statin(i+22);
for b = 1:x
        if b == 1
            statout(b,a+21) = inout(b,12+a) + statin(a+22);
        end
end
% Withdrawals
flowmat(:,a+61,1) = statout(:,a+21)* (24/1000)* parause(a+42);
withtot = withtot + flowmat(:,a+61,1);
% Consumptive use
flowmat(:,a+64,1) = statout(:,a+21)* (24/1000)* parause(a+45);
consutot = consutot + flowmat(:,a+64,1);
end
% Total water withdrawal GW + SW
flowmat(:,70,1) = withtot;

%Total water withdrawn GW
flowmat(:,68,1) = flowmat(:,70,1) * parause(49);
for a = 1:x
    flowmat(a,68,2:4) = flowmat(a,68,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,69,1) = flowmat(:,70,1) - flowmat(:,68,1);
for a = 1:x
    flowmat(a,69,2:4) = flowmat(a,69,1) * parause2(81:83) * 1.382; % tonne/y
end
```

```
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,70,2:4) =  flowmat(:,68,2:4) + flowmat(:,69,2:4);

% Total consumptive water use
flowmat(:,71,1) = -1 * consutot;
% Total water discharged from fuel production processes
flowmat(:,72,:) = flowmat(:,70,:) +  flowmat(:,71,:);

%*********************************************************************
% IRRIGATION WATER USE CALCULATIONS (agricultural and recreational)
%*********************************************************************
% Calculates the factor that relates low intensity urban areas to
% recreational areas. With this, is low intensity area reduces over time so
% the area of recreational area
recAreaFact = statin(3)./ statout(1,9);
% complete variation of irrigated recreational areas, statout(:,25)
statout(:,25) = statout(:,9).*  recAreaFact;

% Conversion factor (0.72374) converts 'millions m3/yr' to 'million gallons
% per day'
% Total water withdrawal GW + SW  (Mgal/day)
re_irr = 0.72374 * statout(:,9).*  recAreaFact * parause(52);
ag_irr = 0.72374 * statout(:,11) * parause(50) * parause(51);

flowmat(:,75,1) = ag_irr + re_irr;

% Total water withdrawn GW
flowmat(:,73,1) = flowmat(:,75,1) * parause(53);
for a = 1:x
    flowmat(a,73,2:4) = flowmat(a,73,1) * parause2(89:91) * 1.382; % tonne/y
end
% Total water withdrawal SW
flowmat(:,74,1) = flowmat(:,75,1) - flowmat(:,73,1);
for a = 1:x
    flowmat(a,74,2:4) = flowmat(a,74,1) * parause2(81:83) * 1.382; % tonne/y
end
% Total water withdrawal GW + SW (COMPOSITION)
flowmat(:,75,2:4) =  flowmat(:,73,2:4) + flowmat(:,74,2:4);

%*********************************************************************
% WATER INPUTS ESTIMATION
%*********************************************************************
% 1. PRECIPITATION
% 2. INTAKE {withdrawals + sewer infiltration + sewer inflow + water imports}
%
% (1) Precipitation
precip = inout(:,1);                                        % mm/year
% Total precipitation (over the whole surface)
flowmat(:,81,1) = (precip.* land) * 0.00072376;        % Mgal/day
% Nutrient Content in precipitation estimated as Wet Nutrient Deposition
% Wet deposition (water component is equal to total precipitation).
% Concentration of nutrients in mg/L and converted to tonne/y assuming
% water density
wet_npc = parause2(41:43) * 1.382;
wet_conv = [1 wet_npc(1) wet_npc(2) wet_npc(3) 0 1382000 1382000];
flowmat(:,81,:) = flowmat(:,81,1) * wet_conv;

% Calculates runoff and evaporation processes
hydpro01

imwt = parause(56) * parause(60);
```

```
% Runoff from impervious areas
ruia = parause(57) * (precip - imwt);                    % mm per year
flowmat(:,76,1) = (ruia.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x % flowmat(:,76,:)
    flowmat(a,76,2:4) = flowmat(a,76,1) * parause2(17:19) * 1.382; % tonne/y
end
% Transfer water from IA to PA
rupa = (1 - parause(57)) * (precip - imwt);              % mm/y
flowmat(:,80,1) = (rupa.* statout(:,26)) * 0.00072376;  % Mgal/d
for a = 1:x
    flowmat(a,80,2:4) = flowmat(a,80,1) * parause2(17:19) * 1.382; % tonne/y
end
% Evaporation from impervious areas (no nutrient value)
flowmat(:,78,1) = -1* (imwt.* statout(:,26)) * 0.00072376;  % Mgal/d

%*********************************************************************
% EVAPORATION FROM PERVIOUS AREAS (Penman's equation)
%*********************************************************************
%
% Air (ambient) tenperature is converted to Kelvin
Tk = statin(31) + 273.15;

% Calculation of Evaporation correction (soil moisture) for each year
% Because this will correct three types of land use we resize matrix
smfactor = etcorrec(precip, statin(27), x);
smfactor =  smfactor * ones(1,3);

% Saturation vapor pressure at air temperature using Goff-Gratch_equation
satvp = -7.90298 * (373.15/Tk - 1) + 5.02808 * log10(373.15/Tk) ...
    - 1.3816e-7 * (10^(11.344 * (1 - Tk/373.15)) - 1) ...
    + 8.1328e-3 * (10^(-3.49149 * (373.15/Tk - 1)) - 1)+ log10(1013.25);

satvp = 10^satvp; % mbar (hPa)
% Another equation for satvp
% satvp = 0.6108 * exp((17.27 * statin(31))/(237.3 + statin(31)));
tvpr = statin(30)./100 * satvp;

% Penman's mass-transfer component (cm/day). Equation 4-21, Dunne & Leopold
pmtc = (0.013 + 0.00016 * statin(32))*(satvp - tvpr);

% Cloudiness Factor
% ccf = bright sunshine hours (h) / maximum possible daylight hours (h)
ccf = parause(61) / 100;


        % Net long wave radiation (cal/cm2/day)
        qlw = 1.17 * (10^-7) * Tk^4 * (0.56-0.08*sqrt(tvpr))*(0.1 + ccf.*0.9);

% Average solar radiation received on a horizontal plane (cal/cm2/day)
asrr = 0.00069*statin(33)^3-0.11146*statin(33)^2-1.45082*statin(33)+ 847.02406;

        % Mean daily solar radiation (cal/cm2/day)
        qs = asrr * (0.803 - ccf.* 0.340 - (ccf.^2).* 0.458);

        % Energy balance matrix:
        Q_Et = zeros(5,4);
        % surface type [water forest grass crops impervious] x columns:
        % [ 1 reflectivity (albedo)
        %   2 Estimation of net radiation (cal/cm2/day)
        %   3 Equivalent depth of Evaporation (cm/day)
```

```
        %   4 Evaporation in mm/year ]

        % Albedo (or Reflectivity): [water forest grass crops impervious]
        % from hydrology book, page 133, excepting for impervious areas
        % (dark roofs 0.20, aged concrete areas 0.25 and paved areas 0.05)
        % for which a ratio roof/concrete/paved of 25:25:50 is assumed (see
        % Araujo Prado 2005 Measurement of albedo)

        impA = 0.25 * (0.20 + 0.25) + 0.50 * 0.05;

        Q_Et(:,1) = [0.07 0.15 0.20 0.20 impA];

        % Estimation of net radiation for different surfaces (cal/cm2/day)
        % [water forest grass crops impervious]
        Q_Et(:,2) = qs * (1 - Q_Et(:,1)) - qlw;

% check that no net radiation is negative (check with water albedo)
if any(Q_Et(:,2) < 0)
    end
else
    emat(6) = 0;
end
        % water density in g/cm3
        wd = 1;
        % water latent heat of vaporization in cal/gr
        wl = 595.0 - 0.51 * statin(31);
        % Equivalent depth of Evaporation for each surface (cm/day)
        % [water forest grass crops -impervious]
        Q_Et(1:4,3) = Q_Et(1:4,2) / (wd * wl);

        % Penman's parameter (dimensionless)
        % Estimated as a function of temperature (table 4-6 hydrology book:
        % )
         pp = 0.69961 * exp(0.05371 * statin(31));

         % Estimation of Evapotranspiration for each surface in mm/year
         % [water forest grass crops -impervious]
         Q_Et(1:4,4) = 3650 * ((pp * Q_Et(1:4,3)) + pmtc) / (pp + 1);


         % Limit evaporation to the amount of precipitation for
         % [forest grass crops] in mm/year
         for luse_et = 2:4
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            if Q_Et(luse_et,4) > precip
            end
         end

% The area fraction of pervious areas in low intensity urban areas is
% estimated from the total area impervious area and the high intensity
% urban area impervious area. {Tarea*%t = Larea*%l + Harea*%h}
Plimperv = (statout(:,26) - parause(147)/100 * statout(:,10))./statout(:,9);

if any(Plimperv > 0.25) == true ;
    end
else
    emat(14) = 0;
end
lperv = statout(:,9).* (1 - Plimperv) ;                % km2
hperv = statout(:,10).* (1 - (parause(147)/100)) ;     % km2
```

```
% Urban forests calculated based on canopy cover for Low and High intensity
% urban areas.
uforests = lperv * parause(145)/100 + hperv * parause(146)/100;
% Grass cover in urban areas is estimated by substracting urban forests
% areas from the pervious area calculated for low and high intensity urban
% area.
lhgrass = (lperv + hperv) - uforests;

% Total GRASS area 'garea' is calculated as the grass area in low and high
% intensity urban areas + clear cut and sparce areas.
garea = lhgrass + statout(:,13) ;   % km2

% Area matrix for each surface type [water forest grass crops impervious]
% Forested Surface (accounts also for urban forests)
% in km2
Et_area = [statout(:,12) (statout(:,8) + uforests) garea ...
    statout(:,11) statout(:,26)];

% Calculation of evapotranspiration (Mgal/day) using area Et_area (km2)
% and evaporation depth Q_Et (mm/year)
% Conversion factor to deal with mm3/year to MGD
AADD = 10^12 * 10^-9 * (1/365) * 264.172 * 10^-6;

Et_vol = zeros(x,4); % [water forest grass crops -impervious]
for a = 1:4
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
    Et_vol(:,a) = Et_area(:,a).* Q_Et(a,4) * AADD;
end
% Evaporation in forest grass crops corrected with a soil moisture
% factor. Surface water evaporates at potential rate (mm/year)
Et_vol(:,2:4) = times(Et_vol(:,2:4),smfactor);

% Evaporation and Evapotranspiration from pervious areas (Mgal/day)
% [water forest grass crops]
flowmat(:,79,1) =  -1 * sum(Et_vol(:,1:4),2);  % flow exiting the system
flowmat(:,79,6:7) =  [flowmat(:,79,1) flowmat(:,79,1)] * 1382000;

% Precipitation (over pervious areas, including water)
perwat = statout(:,27) + statout(:,12);
perprecip = bsxfun(@times,flowmat(:,81,:), perwat./land);
%perprecip = zeros(x,1,nmat);
%for a = 1:x
%    % Mgal/day
%    perprecip(a,:,:) = flowmat(a,81,:).* statout(a,27)./ land;
%end
% check that evapotranspiration is lower than precipitation
if any(abs(flowmat(:,79,7)) > perprecip(:,1,7)) % comparison in mass units
    end
else
    emat(5) = 0;
end
% Effective Pervious Area (epa): water available for either runoff or ground
% infiltration
% This is total water volume received by pervious areas:
% + precipitation
% + irrigation
% + transfer from IA to PA
```

```
% + consumptive use DO, CO, PU (triplec)
% (we substract the N and C removal in septic tanks from DO)
triplec = (flowmat(:,14,:) - flowmat(:,318,:)) + flowmat(:,20,:) + flowmat(:,22,:);
% + Consumptive use MI + LI (flowmat(:,42,:))
% - evaporation in pervious areas (except surface water evaporation)
% (first we convert Et_vol(:,1) into flowmat format)
surw_evap = zeros(x,1,nmat);
surw_evap(:,1,1) = Et_vol(:,1); % Volume MGD of water
surw_evap(:,1,6:7) = [surw_evap(:,1,1) surw_evap(:,1,1)] * 1382000; % tonnes/a of water

% Evapotranspiration from pervious areas (substract open water evaporation)
Perva_evap = flowmat(:,79,:) + surw_evap;
% Left for infiltration (so evaporation is substracted)
epa = perprecip + flowmat(:,75,:) + flowmat(:,80,:) + triplec ...
    + flowmat(:,42,:) + Perva_evap;
% No energy component
epa(:,:,5) = 0;

% Total Ground infiltration
ginfil = zeros(x,1,nmat);
% Parameter: soil infiltration index
ginfil(:,1,:) = parause(59)* epa;

% sewer infiltration, corrected by two parameters:
% (a) fraction of urbanized land (a representation of the coverage of the
% sewerage system)
sinfil = zeros(x,1,nmat);
for a = 1:nmat
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
    l_urban = sum(land_fract(1:x,2:3),2)./sum(land_fract(1,2:3),2);
    sinfil(:,:,a) = l_urban.* ginfil(:,:,a);
end
% (b) sewer infiltration index
sinfil = parause(55) * sinfil;

% Runoff from pervious areas
flowmat(:,77,:) = epa - ginfil;

% Calculation of the fertilization mix (area-wise)
fertmix

tfe = 6; % Types of nutrient supplements (IF, SS, PL, CM, SM, CP)
tpa = statout(:,27); % total pervious area km2
%tro = flowmat(:,77,1); % Total runoff from pervious areas Mgal/d
Areas = [garea statout(:,8)+ uforests statout(:,11)]; % [grass forest crops]
% Forested areas accounts also for urban forests
%*********************************************************************
% FERTILIZER DISTRIBUTION FOR EACH LAND USE TYPE - area terms
%*********************************************************************
% Creates a matrix of the proportion of non-fertilized area for each land
```

```
% use with the size of the alternatives of fertilization
fractnF = zeros(tfe,size(Areas,2));
% [grass forestry crops]
for a = 1:tfe;
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
   fractnF(a,:) = (1-parause(130:132)/100);
end

% Specification of the fertilizer mix (in terms of area proportion) for
% each land use. fe_mix = [IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = a_mix.* (ones(tfe,3) - fractnF);
% fe_mix modified. non-fertilized area fraction is added.
% fe_mix = [nf IF SS PL CM SM CP] x [grass forestry crops]
fe_mix = cat(1,fractnF(1,:),fe_mix);
% depth = [number of years]
fe_mixarea = zeros(tfe+1,3,x); % Pre-allocating
for a = 1:x % each year
    for b = 1:tfe+1; % [nf IF SS PL CM SM CP]
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);
         % Mix of fertilizers as Area km2 [grass forestry crops]
        fe_mixarea(b,:,a) = fe_mix(b,:).* Areas(a,:);

%**********************************************************************
% GROUNDWATER MODELING (ESTIMATION of CONTRIBUTION TO STREAM DISCHARGE)
%**********************************************************************
% Equations from Santhi 2008 "Regional estimation of base flow for the conterminous
% United States by hydrologic landscape regions"
% Groundwater contribution to surface flow: a function of relief and
% available water for infiltration (or runoff, typically P - PET)
% Only water and evapotranspiration associated to pervious areas
Effarea = sum(Et_area,2) * 1000000;
PPET = (epa(:,1) * 365 * 0.0037854 * 1000000./ Effarea) * 1000; % milimiters

% Groundwater contribution to surface flow mm/y
% statin(58) = watershed relief
% PPET = P - PET, similar to effective pervious area, calculated as 'epa'
Gwater = ones(x,1) * (60.43 + 0.2145 * statin(58)) + 0.4283.* PPET;

% Groundwater contribution to surface flow Mgal/d
flowmat(:,95,1) = Gwater.* Effarea /(365 * 0.0037854 * 1000000 * 1000);

% COMPOSITION
for a = 1:x
    flowmat(a,95,2:4) = flowmat(a,95,1) * parause2(89:91) * 1.382; % tonne/y
    flowmat(a,96,2:4) = flowmat(a,96,1) * parause2(89:91) * 1.382; % tonne/y
```

```
% (2) Intake
% 2.1 Withdrawals
% Water withdrawals for public supply (DO + CO + PU + IN)
flowmat(:,6,:) = flowmat(:,7,:) + flowmat(:,15,:) + flowmat(:,23,:) + flowmat(:,24,:);
% Corrected by supply system leakage (distribution and supply losses)
flowmat(:,331,:) = -1 * parause(63) * flowmat(:,6,:)./(1 - parause(63));
flowmat(:,6,:) = flowmat(:,6,:) + (- flowmat(:,331,:));
% Public Supply - Water withdrawals GW
flowmat(:,4,:) = flowmat(:,6,:) * parause(58);
% Public Supply - Water withdrawals SW
flowmat(:,5,:) = flowmat(:,6,:) - flowmat(:,4,:);
% TOTAL from groundwater and surface water
for a = 0:1 % flowmat(:,1,:) and flowmat(:,2,:)
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
    flowmat(:,1 + a,:) = flowmat(:,4 + a,:) + flowmat(:,8 + a,:) + ...
        flowmat(:,16 + a,:) + flowmat(:,25 + a,:) + flowmat(:,39 + a,:)...
        + flowmat(:,57 + a,:) + flowmat(:,68 + a,:) + flowmat(:,73 + a,:);
end
% Export fresh water assumed from SW source
flowmat(:,2,:)= flowmat(:,2,:) - flowmat(:,284,:);

% TOTAL GW + SW withdrawls (includes export/import of fresh water)
% Water exports are negative, flowmat(:,284,1) < 0
flowmat(:,3,:) = flowmat(:,1,:) + flowmat(:,2,:);

% 2.2
% Calculation of sewer infiltration was done in 'hydpro01'...(sinfil)
% Sewer inflow is a function of impervious runoff from low and high
% intensity urban areas only.
sinflow = flowmat(:,76,:) * parause(54);
% Inflow/Infiltration to sewer networks
flowmat(:,82,:) = sinfil + sinflow;
flowmat(:,320,:) = sinflow;
flowmat(:,321,:) = sinfil;

%*********************************************************************
% WATER STORAGE CALCULATIONS (only surface reservoirs)
%*********************************************************************
% Estimation of water storage capacity
% We use a correlation valid only for LAKE LANIER (case study UCW)
reflevel = 313.93;                      % m-msl
sini = [statin(34) inout(1:x-1,3)']' - reflevel;
send = inout(:,3) - reflevel;
% Calculation of storage capacity
capini = (196045 * exp(sini.*0.1102));    % Mgal
capend = (196045 * exp(send.*0.1102));    % Mgal
% Calculation of the differential in storage for each year
capdiff = capend - capini;
% Calculation of the differential in storage Mgal/day
statout(:,28) = capdiff./ 365;
% Composition of Water storage, fisrt flow equivalent to the change in
% storage
flowmat(:,286,1) = statout(:,28);
for a = 1:x
    flowmat(a,286,2:4) = statout(a,28) * parause2(81:83) * 1.382; % tonne/y
end
%*********************************************************************
% WATER OUTPUT CALCULATIONS
```

```
%************************************************************************
% 3. HYDROLOGY
% Ground infiltration was estimated in 'hydro01' (ginfil), as well as sewer
% infiltration (sinfil). Correction is made to include:
% - leaching from land (grass, crop, forest)in leaching.m
% - leaching from landfill and composting in wastems.m
% flowmat(:,83,:)
% The water left after sewer infiltration is available for
% baseflow recharge and groundwater flow (to other watersheds).
flowmat(:,83,:) = -1 * (ginfil - sinfil);

%************************************************************************
%********** (these two flows are corrected in runoff_pa) *****************
% Total runoff generated from IA + PA
flowmat(:,84,:) = flowmat(:,76,:) + flowmat(:,77,:);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,:) = -1 * (flowmat(:,84,:) - sinflow);
%************************************************************************
% Evapotranspiration from PA + IA
flowmat(:,85,:) = flowmat(:,78,:) + flowmat(:,79,:);
% Evaporation from industry + PG + FP
flowmat(:,86,:) = flowmat(:,30,:) + flowmat(:,60,:) + flowmat(:,71,:);
% Total Evaporation
flowmat(:,87,:) = flowmat(:,85,:) + flowmat(:,86,:);

% Water/Wastewater exported/imported to another basin
% A negative value represents an outbound flow: export.
flowmat(:,284,:) = inout(:,4) * [1 parause2(81:83)'*1.382 0 1382000 1382000];  % Water
flowmat(:,285,:) = inout(:,5) * [1 parause2(1 : 3)'*1.382 0 1382000 1382000];  % Wastewater
% Energy. Based on Reynolds/Richards. Unit Operations and Processes in
% Environmental Engineering (table 5.2) we expect a TS/BOD ratio between
% 3.0-4.5. We use 4.0
flowmat(:,285,5) = flowmat(:,285,4) * 4 * parause2(4); % kWh/y
% Correction to convert BOD into Total Carbon (TC) for only those years
% where wastewater is imported or exported
if sum(flowmat(:,285,4),1) ~= 0
end
% Watershed total drainage (river drainage + ground water drainage)
% precip - ET - Storage + imp/exp W + imp/exp WW (exports are negative)
flowmat(:,88,:) = -1 * (flowmat(:,81,:) + flowmat(:,87,:) ...
    - flowmat(:,286,:) + flowmat(:,284,:) + flowmat(:,285,:));

% 4. DISCHARGES (to surface water)
% no discharges to ground water are considered (if any)
% 4.1 Discharges through WWTPs
% Total wastewater returned from DO, CO, and PU use (withdrawals - consumptive)
dom = flowmat(:,11,:) - flowmat(:,14,:);
com = flowmat(:,19,:) - flowmat(:,20,:);
pub = flowmat(:,21,:);
% Calculation of water to wasetwater treatment (water)
maxW2wwtp = dom + com + pub;
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(1:3)'*1.382;
if x > 1
    end
end
maxW2wwtp(:,:,2:4) = diag(maxW2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
```

```
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxW2wwtp(:,:,5) = maxW2wwtp(:,:,4) * 4 * parause2(4); % kWh/a
% Water weight in tonnes/a
maxW2wwtp(:,:,6) = maxW2wwtp(:,:,1) * 1382000;
maxW2wwtp(:,:,7) = maxW2wwtp(:,:,1) * 1382000;

% Total water returned from INDUSTRIAL use (withdrawals - consumptive)
maxI2wwtp = flowmat(:,28,:) - flowmat(:,29,:) + flowmat(:,30,:);
% Calculation of species
% Corrected to include changes in consumption over time (columns 2 to 5)
% Column 4 is in BOD (tonnes/a); converted to total carbon in 'comp_wwtp.m'
paraTemp = ones(x,1) * parause2(49:51)'*1.382;
if x > 1
    end
end
maxI2wwtp(:,:,2:4) = diag(maxI2wwtp(:,:,1),0) * paraTemp;
% Energy calculated from TS (based on Reynolds/Richards Table 5.2):
% Typical TS/BOD ratio between 3.0-4.5. We use 4.0
maxI2wwtp(:,:,5) = maxI2wwtp(:,:,4) * 4 * parause2(52); % kWh/y
% Water weight in tonnes/a
maxI2wwtp(:,:,6) = maxI2wwtp(:,:,1) * 1382000;
maxI2wwtp(:,:,7) = maxI2wwtp(:,:,1) * 1382000;

% Industrial wastewater to WWTP (into flowmat matrix)
flowmat(:,90,:) = maxI2wwtp(:,:,:);
% Correction to convert BOD into Total Carbon (TC)
bodML = flowmat(:,90,4)./ (flowmat(:,90,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
tcML = tcbod(bodML,2,2); % TC mg/L
flowmat(:,90,4) = tcML.* flowmat(:,90,1) * 1.38200; % TC tonne/y

% Estimation of flow and composition of WWTP influent based on water
% consumption, typical concentrations, source separation (SST) and
% reduction of water use
comp_wwtp

TripleI = flowmat(:,285,:) + flowmat(:,82,:);  % TC tonne/a
% Conversion from m3/s to MGD
% TripleI(:,:,1) = TripleI(:,:,1)./(1000000 * 0.0037854 / (24 * 3600));
TripleI(:,:,4) = TripleI(:,:,4)./ (TripleI(:,:,1) * 1.38200); % TC mg/L
% TC to BOD mg/L
triBOD = tcbod(TripleI(:,:,4),1,2); % BOD mg/L
TripleI(:,:,4) = triBOD.* TripleI(:,:,1) * 1.38200; % BOD tonne/a
TripleI(:,:,5) = 0;

% Aggregation of WWTP clients (no SST implemented)
Rawsew_nosst = maxW2wwtp + maxI2wwtp + TripleI; % Carbon as BOD tonne/a

% patch
% Organic waste (food) is managed through the sewage system (cow)
Rawsew_nosst = Rawsew_nosst + flowmat(:,330,:);

% Carbon
% **** BOD is converted to Total C ***********
infBOD = Rawsew_nosst(:,:,4)./(Rawsew_nosst(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
Rawsew_nosst(:,:,4) = infTC.* Rawsew_nosst(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
Rawsew_nosst(:,:,6) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y
% Water Mass
```

```
Rawsew_nosst(:,:,7) = Rawsew_nosst(:,:,1) * 1382000; % tonne/y

%**********************************************************************
% INFLUENT to WWTPs
% *** CORRECTED WITH SST (urine separation)
%**********************************************************************
% Population that does contribute to WWTP influent, i.e. 'sew_pop',
% 'sep_pop' is the population using septic tanks.
% Population connected to sewer network
sew_pop = statout(:,2) - sep_pop;
% portion of population connected to sewer network
pro_pop = sew_pop./statout(:,2);

% Urine volumen that can be separated per person per day based on
% Source-Separation Toilet (sst) efficiency 'parause(97)'
usep_v = (parause(95)./1025)* 264.172* parause(97); % gpd per cap

% (a) The sewage composition changes as urine separation is implemented,
% but only associated to those connected to the sewer network (pro_pop).
% (b) Effluent from the struvite process is added to the sewage composition
% Pre-allocating matrix space
INF_wwtp = zeros(x,1,nmat); % [W N P C E M WaterM]
INF_d_W2 = zeros(x,1,nmat); % [W N P C E M WaterM]

% First the difference created by the implementation of urine separation is
% computed:
INF_d_W2(:,:,1) = (sew_pop./1000) * (nstwuse + usep_v - sstwuse) * parause(98);
% Composition in tonne/y.
for a = 2:7 % [N P C E M WM]
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
    INF_d_W2(:,:,a) = flowmat(:,131,a).* pro_pop(:);
end
% Influent to WWTP corrected
INF_wwtp = Rawsew_nosst - INF_d_W2 + stru_relea;
% stored in flowmat matrix
flowmat(:,96,:) = INF_wwtp;

% Wastewater from DO, CO, and PU corrected by the use of urine separation
% Aggregation of WWTP clients (no SST implemented)
% Carbon as BOD tonne/a ALSO corrected by organic waste (cow)
INF_DOM2 = maxW2wwtp + flowmat(:,330,:);
% Carbon **** BOD is converted to Total C ***********
infBOD = INF_DOM2(:,:,4)./(INF_DOM2(:,:,1) * 1.38200); % BOD mg/L
% BOD to TC mg/L
infTC = tcbod(infBOD,2,2); % TC mg/L
INF_DOM2(:,:,4) = infTC.* INF_DOM2(:,:,1) * 1.38200; % TC tonne/y
% Total Mass (mostly water)
INF_DOM2(:,:,6) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Water Mass
INF_DOM2(:,:,7) = INF_DOM2(:,:,1) * 1382000; % tonne/y
% Correction of original sewage from DO, CO, and PU
flowmat(:,89,:) = INF_DOM2 - INF_d_W2;

% Estimation of WWTP emissions
% we place here a loop to correct for the SUPERNATANT flow that is routed
% to the WWTP headwaters (influent flow)
```

```
% Matrix allocation for SUPENATANT flow
SN_flow = zeros(x,nmat); % years x 1 x [W N P C E M]
% Matrix allocation for emissions from biological treatment
biot_air = zeros(x,3); % years x [N P C]
% Estimation of WWTP emissions and discharges
emi_wwtp

flowmat(:,91,2) = INF_wwtp(:,2) * (1 - parause(118));
% Phosphorus
flowmat(:,91,3) = INF_wwtp(:,3) * (1 - parause(107));
% Carbon, based on BOD removal. Includes annual reduction of BOD regulation
flowmat(:,7,1) = (pspo./1000).* (chdow - parause(98)*(nstwuse - sstwuse));
for a = 1:x
    flowmat(a,7,2:4) = flowmat(a,7,1) * parause2(89:91) * 1.382; % tonne/y
end
if parause(234) == 0
    effBOD = infBOD.* (1 - parause(105)); % Effluent BOD mg/L
        end
    end
end
% **** BOD is converted again to TC ***********
%
% TC/BOD effluent in mg/L
% BOD to TC mg/L
effTC = tcbod(effBOD,2,1);
% Water flow approximated to total WWTP water input
% TC effluent (C tonne/a), INF_wwtp (MGD)
flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;

%**********************************************************************
% Algae growing from WWTP effluent
%**********************************************************************
% Algae cells (on a dry basis) are assumed an atomic formula similar to
% CAyHAzOAtNAxPAt. Hydrogen is calculated from the oxygen molar number from
% reference (Stumm and Morgan 1981).
% [AlgaeCells Lipids Biomass x years] x [C H O N P S Ash], moles
% First two rows are moles of Algae cells and Lipids, the remaining rows
% are the biomass for each year.
AUlt = zeros(2+x,7);
AUlt(1,:) = [12*parause2(35) 1*parause2(36)*2.4 16*parause2(36) ...
    14*parause2(33) 31*parause2(34) 32*0 0];
%
% Total atomic MASS of algae cells (kg/kmol)
TAx = sum(AUlt(1,:));
% Only N, P, C
aAl = [AUlt(1,4) AUlt(1,5) AUlt(1,1)];
% Assumed ash content of algae biomass (percentage of dry mass)
a_A = 8; % Reference \citep{Zhu_2010}
% Matrix of necessary nutrients to achieve the desired algae growth
T_algae = zeros(x,1); % total mass of dry algae
M_algae = zeros(x,1); % total number of mols of dry algae
FertAlg = zeros(x,1,nmat);

% Algae process
if parause(65) > 0
    AlgaeW

if x < 2
    What1(:,:) = flowmat(:,91,2:4);
    What = What1';
end
```

```
% What is N, P, C
%What(:,:) = flowmat(:,91,2:4);
AvMoles = What(:,:)./(ones(x,1) * [14 31 12]);% molar
AvMoles = AvMoles./(ones(x,1) * aAl);%

% Limiting nutrient is one with the lowest number
[val,NutInt] = min(AvMoles,[ ],2);

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% Also influenced by the nutrient recovery factor (efficiency)
for a = 1:x
    T_algae(a,:) = flowmat(a,91,NutInt(a)+1) * parause(65) * parause(66) * TAx./aAl(NutInt(a))';
end
% Calculation of nutrient mass in produced algae
% number of (metric tonnes) mols of algae cells
M_algae = T_algae./ TAx;
MratioAlgae = [parause2(33) parause2(34) parause2(35)];
M_nutrients = M_algae * MratioAlgae;
% convert moles to mass N P C
AlMass = M_nutrients.*(ones(x,1) * [14 31 12]);
% (first we convert AlMass into flowmat format)
AlMassF = zeros(x,1,nmat);
AlMassF(:,1,2:4) = AlMass(:,:);

%% This was moved inside the IF Clause in 02/03/2014. Is was right after.
% Effluent of Algae process (WWTP Effluent minus Algae retention)
% As an EXERCISE, we say C comes from the effluent, as opposed to CO2
% flowmat(:,91,3) - AlMassF(:,:,3)
effAlgae = flowmat(:,91,:) - AlMassF(:,:,:);
% **** TC is converted to BOD ***********
% TC in mg/L
eff_C_mgL = effAlgae(:,:,4)/(INF_wwtp(:,1) * 1.382);
% TC to BOD mg/L
effAlgae_BOD = tcbod(eff_C_mgL,1,1);
% Water flow approximated to total WWTP water input
% TC effluent C tonne/y
%flowmat(:,91,4) = effTC.* INF_wwtp(:,1) * 1.38200;
end

%*********************************************************************
% Emissions to air
%*********************************************************************
% - SEWER NETWORK
% It is assumed that the known concentration of raw sewage corresponds to
% the inffluent of a typical wastewater treatment plant. Therefore,
% emissions from the sewer network are backcalculated as lost mass before
% reaching the WWTP inlet
sewer_air = zeros(x,3); % years x [N P C]
% Nitrogen
sewer_air(:,1) = 0;
% Phosphorus
sewer_air(:,2) = 0;
% Carbon
sewer_air(:,3) = 0;

% - BIOLOGICAL TREATMENT
% Next: calculation of Nitrogen emissions (N2 + NH3 + N2O)
% ASA available N for sludge and air emissions
ASA = INF_wwtp(:,2) * parause(118);
% AFA available N for air emissions
```

```
AFA = ASA * (1 - parause(106));
% AN2 available N for N2 emissions (only)
AN2 = AFA * (1 - parause(116) - parause(117));

biot_air(:,1) = -1 * AFA;
% Phosphorus
biot_air(:,2) = 0;

% Carbon, calculated stoichiometrically, 1 mol O2 generates 1 mol CO2
sludact

Cy = parause2(15);
Hz = parause2(14);
Ot = parause2(16);
Nx = parause2(13);
% Total atomic mass of sludge cells
Tx = 12*Cy + 1*Hz + 16*Ot + 14*Nx;

%*********************************************************************
% Activated sludge reactor
%*********************************************************************
% eq 15.92 Reynolds/Ricards book. We use the prefix 'AS' in reference to
% 'Activated Sludge'. Simulated as a conventional reactor
% Crosscheck for a conventional reactor
% (a) food-to-microbes (0.2-0.4 kg BOD/Kg MLSS-day)
% (b) space loading (0.3-0.6 kg BOD/day-m3) parause(114)
% hypothetical reactor volumen (m3). In this case we don't use the
% corrected value of inlfuent BOD but rather the original BOD, assuming
% that the latter is the design BOD of the WWTP
AS_Rvo = INF_wwtp(:,1) * 3.7854117834 * 1000000 * ...
    parause2(3)/(1000000 * parause(114));
% Substrate (BOD5) removal rate (tonnes BOD5/day). Actual values the BOD (corrected for
% technology implementation)
AS_Sre = INF_wwtp(:,1) * 3.7854117834 * 1000000 .* ...
    (infBOD - effBOD) / 1000000000;
% Total Mass of Volatile solids in the reactor (tonne MLVSS)
AS_Xvs = AS_Rvo * 1000 * parause(112) * parause(113) / 1000000000;
% Volatile suspended solids produced (tonne MLVSS/year)
% check for a positive number
AS_Pvs = (AS_Sre.* parause(111) - AS_Xvs * parause(110)) * 365;
% Mean cell resident time (days)
% check (usually 10-30 days)
AS_Cre = (AS_Xvs./AS_Pvs) * 365;
% Total solids (tonne MLSS/year)
AS_Tss = AS_Pvs / parause(113);

if any(AS_Pvs < 0)
    end
end
% Oxigen (O2) demand in tonne/a. See book
% eq 15.93 Reynolds/Ricards book, no nitrification
% only for the purposes of estimating the amount of carbon that is
% realeased via endogeneous respiration
%AS_Ovs = (AS_Sre.* parause(109) + AS_Xvs * parause(108)) * 365;
AS_Ovs = AS_Xvs * parause(108) * 365;
biot_air(:,3) = -1 * AS_Ovs * 12/32;

% - TOTAL EMISSIONS TO AIR
flowmat(:,130,2:4) =  biot_air + sewer_air;

%*********************************************************************
```

```
% Solid waste - SEWAGE SLUDGE GENERATION
%*********************************************************************
% - Untreated Sewage Sludge (FEED TO SLUDGE DIGESTERS)
% ==================================================
% Water and Solids (assumed 4% of solids after thickening)
% AS_Tss calculated in 'sludact.m'
flowmat(:,129,6) = AS_Tss / 0.04;
flowmat(:,129,1) = (flowmat(:,129,6) - AS_Tss) / 1382000;
flowmat(:,129,7) = flowmat(:,129,6); % Approximated to 100% water
% Nitrogen
flowmat(:,129,2) = INF_wwtp(:,2)  + biot_air(:,1) - flowmat(:,91,2);
% Check that flowmat(:,129,2)./INF_wwtp(:,2) is near 15%, based
% on numbers reported by Sonesson 2004, Mechanical + Chemical + Biological
% with N removal (Table 2).
% Phosphorus
flowmat(:,129,3) = INF_wwtp(:,3) * parause(107);
% Carbon
flowmat(:,129,4) = INF_wwtp(:,4) + biot_air(:,3) - flowmat(:,91,4);
%INF_wwtp(:,4) - flowmat(:,91,4)
% Energy is not estimated
%- Sewage Sludge Digestion Process
%==================================
% Includes calculations of biogas, treated sludge, and supernatant
sluddig

SD_par = [ 0.45 0.55
           0.60 0.70 ];
% Sludge digestion temperature level (1 meso 2 Thermo)
phil = 2;

% Volatile solids destroyed in digestion (tonne MLVSS/year) is a function
% of the Volatile solids produced in the Activated Sludge Reactor
SD_des = AS_Pvs * SD_par(1,phil);
% Total solids after digestion (tonne MLSS/year)
% check, Total solids destruction = 1 - SD_Tss./AS_Tss (typically 35-50);
SD_Tss = AS_Tss - SD_des;

% Calculation of biogas. Based on CyHzOtNx
% See notebook pag. 87 for details on equations
% Carbon mass released as gas (tonne/a). It is assumed that CH4 and CO2 are
% the only relevant components of BIOGAS from sewage sludge
SD_car = SD_des * (Cy*12)/Tx;
% Mass of Methane released (tonne/a)
SD_met = SD_car./(0.750 + 0.273 * (2.75 * (1 - SD_par(2,phil))/SD_par(2,phil)));
% Volume of Methane released (m3/a): equation uses mass in tonnes
SD_metV = calmass(1,0,16,SD_met,1); % [P,T,MW,mass or vol,mod]
% check, Methane production rate is reported to be between 1000 m3
% (Kabouris 2009) and 400 m3 (Song 2004) per tonne of VS destroyed
% SD_metV./SD_des
%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,277,2) = 0;
flowmat(:,277,3) = 0;
flowmat(:,277,4) = SD_car;
flowmat(:,277,5) = SD_metV * 10.343; % kWh/a
```

```
%************************************************************************
% TREATED SLUDGE FLOW
%************************************************************************
% Calculation of Water and solids in the treated sludge flow.
% After dewatering, solids are reduced to 14-22% (Reynolds book)
% We assume 20% solids flowmat(:,129,1)
flowmat(:,289,6) = SD_Tss / 0.20;
flowmat(:,289,7) = flowmat(:,289,6) - SD_Tss;
flowmat(:,289,1) = flowmat(:,289,7)./ 1382000;
flowmat(:,289,2) = flowmat(:,129,2);
% check N concentration (mass fraction)
% flowmat(:,289,2)./SD_Tss
flowmat(:,289,3) = flowmat(:,129,3);
% check P concentration (mass fraction)
% flowmat(:,289,3)./SD_Tss
% Carbon (corrected by VS destruction)
flowmat(:,289,4) = flowmat(:,129,4) - SD_car;
% check C concentration (mass fraction)
% flowmat(:,289,4)./SD_Tss % dry mass
% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
m_A = 35; % Ash content as a percentage
m_S = 1; % Sulphur content as a percentage
num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
m_C = num_moles * Cy * 12;  % Carbon content as a percentage
m_H = num_moles * Hz * 1;   % as a percentage
m_O = num_moles * Ot * 16;  % as a percentage
m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;

% effluent water flow after Treated Sludge water content correction
flowmat(:,91,1) = INF_wwtp(:,1) - flowmat(:,289,1);

% check concentration TN in mg/L
% Flow in m3/s
% flowmat(:,91,2)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 3.0 (0.6-4.6) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
% check concentration TP in mg/L
% flowmat(:,91,3)./(flowmat(:,91,1)* 22.8245 * 1.382)
% 0.4 (0.1-0.7) Ref: Mines Jr 2004 Assessment of AWT metro Atlanta
%************************************************************************
% ENERGY USAGE BY WWTP
%************************************************************************
% Based on F. Jiang modeling for WWTP with or without urine separation
% (1) with complete mix scenario, the energy consumption of WWTP is 0.33 kwh/m3.
% (2) with 100% urine separation, the energy consumption is reduced to 0.31 kwh/m3.
% 1 US Gallon = 0.00378541178 Cubic Meters
% Converted to: (1) 0.00125 kWh/gal (2) 0.00117 kWh/gal
% Energy is reduced by about 10% if all urine is removed from the
% sewage, so energy is calculated as proportional to SST implementation
%*****************************************
% Energy use Wastewater treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
```

```
% varies from 800  - 3,500 kWh/MG
% Procedure divides by the number of water treatment plants 'statin(59)'
% and later totalizes
ene_wwtp =  parause(225) + ...
            parause(226) * log(INF_wwtp(:,1)/statin(59)) + ...
            parause(227) * log(infBOD) + ...
            parause(228) * log(effBOD) + ...
            parause(236) * log(100 * (1 - parause(64))) + ...
            parause(237) * 1; % last term assumes nutrient removal
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_wwtp2 = exp(ene_wwtp) / 11.1;
% Primary energy ene_wwtp2 = exp(ene_wwtp)*1000*0.00029307107017;
% mean(flowmatRSA(:,332,5))
% Assuming 10% in energy requirement reduction (from 0 to 1 of urine
% separation technology implementation). Based on Feng Jiang's simulations
flowmat(:,332,5) = ene_wwtp2 * (1 - 0.10 * parause(98)) * statin(59);

% check ratio kWh/MG in electricity terms
% (ene_wwtp2 * statin(59)./(INF_wwtp(:,1) * 365))
%*********************************************
% Energy use water treatment
% From file 'Water wastewater energy benchmark Final Report.pdf'
% varies from 300  - 2,400 kWh/MG (average 1700)
% Procedure divides by the number of water treatment plants 'statin(56)'
% and later totalizes
ene_w = parause(229) +  ...
        parause(230) * log(flowmat(:,6,1)* 1000 / statin(56)) + ...
        parause(231) * log(statin(55)* flowmat(:,6,1) / statin(56)) + ...
        parause(238) * log(statin(57)) + ...
        parause(239) * log(statin(58) * 3.281 / statin(56));
origMGD = flowmat(:,6,1);
%ene_w = parause(229) +  parause(230) * log(flowmat(:,6,1)) + ...
%    +  parause(231) * log(water_hp);
% In electricity energy terms, from kBTU to kWh/y (process energy)
% Benchmark report indicates (table 3.1) a ratio kBTU to kWh of electricity
% of 11.1
ene_w2 = exp(ene_w) / 11.1;
% Primary energy ene_w2 = exp(ene_w) * 1000 * 0.00029307107017;
% Total energy (electricity) for all water treatment works
flowmat(:,333,5) = ene_w2 * statin(56);
% mean(flowmatRSA(:,333,5))
% Energy requirements increase proportionally with the amount of
% groundwater involved in water supply.
% check ratio kWh/MG in electricity terms
% (ene_w2 * statin(56)./(origMGD * 365))
%*********************************************************************
% Consistency verification
%*********************************************************************
% Verification that Urine's nutrient contribution to the sewer network
% does not exceed the amount of nutrient calculated by sewage typical
% concentration plus the emissions (losses) through the sewer network
% A minimum of 15% is assumed to come from sources other human excretion
if any(flowmat(:,127,2).* pro_pop.*1.15 > Rawsew_nosst(:,2) + sewer_air(:,1))
    end
else
    emat(8) = 0;
end
if any(flowmat(:,127,3).* pro_pop.*1.15 > Rawsew_nosst(:,3) + sewer_air(:,2))
```

```
    end
else
        emat(12) = 0;
end
if any(flowmat(:,127,4).* pro_pop.*1.15 > Rawsew_nosst(:,4) + sewer_air(:,3))
    end
else
        emat(13) = 0;

% 4.2 Discharges from (IN, PG, and FP) and (MI+LI )
% Other water returns to surface sources
indu    = flowmat(:,29,:) + flowmat(:,61,:) + flowmat(:,72,:);
mili    = flowmat(:,41,:) - flowmat(:,42,:);
flowmat(:,92,:) = indu + mili;

% TOTAL WATER RETURNS: WWTPs AND INDUSTRY
flowmat(:,93,:) = -1 * (flowmat(:,91,:) + flowmat(:,92,:));
% flowmat(:,93,:) will be corrected in wastems.m based on the struvite
% process releases
% Total water consumptive use (evaporation IN + PG + FP) calculated
% previously in flowmat(:,86,:)
% Total water consumptive use (irrigation + infiltration + leakage)
% Not evaporation
% DO, CO, PU, LI + MI, Irrigation (farming), water supply leakage
% It includes septic tanks
flowmat(:,94,:) = flowmat(:,14,:) + flowmat(:,20,:) + ...
    flowmat(:,22,:) + flowmat(:,42,:) + flowmat(:,75,:) + (-flowmat(:,331,:));

% MODULE Forestry sector
forestrys;

swcmtomt = 0.6; % [tonnes / m3]
% Conversion factor board feet (bf) to air dried weight in metric tons
swbftomt = 0.0012; % [tonnes /bf]

% Calculation of live tree inventory
% Net growth rate of biomass in timberland (forestry) in tonnes/a
% Growth minus removals
STgrowth = statout(:,31) * (parause(69) - parause(70)) * 100 * swcmtomt;
% Total biomass (inventory) in timberland
statout(1,32) = statin(36) * swcmtomt * statout(1,31);  % in tonnes/a
if x > 1
    end
end
% ********************************************************
% Calculation of live tree removals
% ********************************************************
%******* REMOVALS **********
% Removals calculated in metric tonnes per annum (green) from m3/a
flowmat(:,97,6) = statout(:,31) * parause(70) * 100 * swcmtomt;

% Nutrient and Energy content based on amount of bark and wood
flowmat(:,97,2:5) = flowmat(:,97,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))';
% Total mass corrected for moisture content
flowmat(:,97,7) = flowmat(:,97,6) * parause2(133);
flowmat(:,97,1) = flowmat(:,97,7) /1382000;

%******* RESIDUE **********
% Calculation of logging residue (some left onsite, some used as fuel.
% Estimated as a portion of removals.
```

```
flowmat(:,98,:) = flowmat(:,97,:) * parause(71) ; % Softwood

% Logging residue sent to waste management
flowmat(:,144,:) = flowmat(:,98,:) * (1 - parause(72)); % tonne/y

% Total ouput of roundwood products (sawlogs, veneer, others)
flowmat(:,99,6) = -1 * parause(76) * 1000; % tonne/y
% Composition in dry basis
flowmat(:,99,2:5) = flowmat(:,99,6) * parause2(129:132)' * (1 - parause2(133));
% Water flow
flowmat(:,99,7) = flowmat(:,99,6) * parause2(133);
flowmat(:,99,1) = flowmat(:,99,7) /1382000;

%*********************************************************************
% CALCULATIONS ASSOCIATED TO FORESTS - NON-TIMBER LAND
%*********************************************************************
% Changes in pools {biomass (B) + dead organic matter (DOM) + soil organic
% matter (SOM)} are estimated for non-timber land
notimber

noTarea = (statout(:,8) - statout(:,31) + uforests) * 100;     % hectares
noTgrowth  = noTarea * parause(121);               % tonne d.m./y
% Initial biomass value is taken from 2006 IPCC Guidelines for National
% Greenhouse Gas Inventories, Vol 4 Chapter 4, Table 4.7. The rest is
% calculated as {the initial value -/+ loss/gain land - mortality + growth }
% The first two are estimated by correcting the initial biomass stock by
% the change in non-timber land. The Second two terms assumes that
% mortality is a funcion of growth.
noTbiomass = zeros(x,1);
noTbiomass(1) = noTarea(1) * parause(122);               % tonne d.m.
if x > 1

%*********************************************************************
% ABOVE- and BELOW-GROUND INVENTORY
%*********************************************************************
% - all forested land
Invforest = (noTbiomass + statout(:,32) + statout(:,33)) * (1 + parause(123));

%*********************************************************************
% CALCULATION OF CARBON POOLS
%*********************************************************************
% - BIOMASS GROWTH
%  all forested land, in tonnes/a of d.m.
statout(:,35) = (noTgrowth + STgrowth/(1-parause(120))) * (1 + parause(123));

fBIO = statout(:,35) * parause2(131); % tonne C/y

% Grassland: standing biomass is assumed that does not change (constant
% inventory) based on the assumption that grassland are harvested or mowed
% constantly. Therefore, we assume that grass growth rate is equal to yard
% waste.
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site.
flowmat(:,119,:) = flowmat(:,111,:)/(1 - parause(92) - parause(137));

% Yard waste and grass left on site
ywLeft = flowmat(:,119,:).* parause(137);

gBIO = flowmat(:,119,4); % tonne C/y
```

```
% Cropland:
% That is the increase in biomass stocks in a single year is equal to the
% biomass losses from harvest and mortality in that same year, thus no net
% accumulation of biomass carbon
% stocks
cBIO = zeros(x,1);  % tonne C/y
% TOTAL BIOMASS CARBON STOCK CHANGE
TBIO = fBIO + gBIO + cBIO;

% - DEAD ORGANIC MATTER (LITTER AND DEAD WOOD)
% Flow of transfered from biomass to DOM pool in tonne d.m./y
Lmortality = statout(:,35) * parause(120);
% DOM change in forests: Lmortality + residue after harvesting (tonne/y)
fDOM = Lmortality * parause2(131) + flowmat(:,98,4); % tonne C/y
% DOM change in grassland:
gDOM = zeros(x,1);
% DOM change in cropland: See the reasoning explained for Biomass growth
cDOM = zeros(x,1);
% TOTAL DOM CARBON STOCK CHANGE
TDOM = fDOM + gDOM + cDOM;

% - CARBON SOIL STOCK CHANGE - All land: FORESTS, CROP, GRASS
% Based on 2006 IPCC guidelines (Volume 4, chapter 2)
% Areas has columns [grass forest crops]. Calculated in 'fertmix.m'
% 'forests' in 'Areas' includes urban forests and non-timber land
% Flu is land use stock change factor for each land use
% Fmg is management regime stock change factor for each land use
% Fin is input of organic matter stock change factor for each land use
% **** WARNING - THESE STOCK CHANGE FACTORS (SCF) HAVE NOT BEEN VERIFIED
% FOR ANY SPECIFIC SITE - DEFAULT VALUES ARE BEING USED
% See Volume 4, table 6.2 for grassland SCF
% See Volume 4, table 5.5 for cropland SCF
%    [grass forest crops]
Flu = [1.00 1.00 0.69];
Fmg = [1.10 1.00 1.08];
Fin = [1.00 1.00 1.05];
% 'D' corresponds to the time dependece of SCF - DEFAULT VALUE USED
D = 20; % years
SOCmat = zeros(x,3); % [years] x [grass forest crops]
for a = 1:x
    SOCmat(a,:) = parause(119) * (Flu.*Fmg.*Fin).* Areas(a,:) * 100;
end
% Value of Soil Organic Carbon for each year
SOC = sum(SOCmat,2);
% Estimated change of SOC in mineral soils, tonne C/y
DCmin = zeros(x,1);
for a = 1:x-1
end
% Losses of Carbon from drained soils. factors in tonne C/ha*y, while
% Lorganic in tonne C/y
%           [grass       forest         crops]
Lorgf = [parause(103) parause(102) parause(104)] ;
Lorganic = zeros(x,1);
for a = 1:x
    Lorganic(a,:) = sum(Lorgf.*Areas(a,:));
end
% Estimated ANNUAL CHANGE IN CARBON STOCKS IN SOILS, grass+forest+grass
% change of SOC in mineral soils - Losses of Carbon from drained soils
% tonne C/y (negative values are losses)
DCS = DCmin - Lorganic;
```

```
% net photosynthesis/respitation as carbon in tonne/y
% This is a measure of C sequestration
flowmat(:,148,4) = TBIO + TDOM + DCS;

%*********************************************************************
% CALCULATION of SAWMILLS input and output
%*********************************************************************
sawmill

smcap0 = statin(38) * 1000000 * swbftomt;
% Sawmill increase in capacity in metric tons
smcap1 = inout(:,16) * 1000000 * swbftomt;
%
for a = 1:x
       if a == 1
           statout(a,34) = smcap1(a) + smcap0;
       end
end
% Estimation of the maximum input capacity of sawmills based on the throuput
% and the amount of byproducts (tonne/y). The flow of wood to sawmills is
% assumed to be equal to their capacity.
% input capacity includes wood and bark
flowmat(:,101,6) = statout(:,34)./parause(74);
% Nutrient and Energy content based on amount of bark and wood
flowmat(:,101,2:5) = flowmat(:,101,6) * (parause2(129:132) * ...
    (1 - parause(75)) + parause2(134:137) * parause(75))' * (1 - parause2(133));
flowmat(:,101,7) = flowmat(:,101,6) * parause2(133);
flowmat(:,101,1) = flowmat(:,101,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL OUTPUTS
%*********************************************************************
% Final product from sawmills to "Local Consumption" (tonne/y)
flowmat(:,103,6) = statout(:,34);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,103,2:5) = flowmat(:,103,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,103,7) = flowmat(:,103,6) * parause2(133);
flowmat(:,103,1) = flowmat(:,103,7) /1382000;

%*********************************************************************
% CALCULATION OF SAWMILL RESIDUE
%*********************************************************************
% Residue of sawmills
flowmat(:,102,:) = flowmat(:,101,:) - flowmat(:,103,:);
%flowmat(:,101,2)./flowmat(:,101,6)
%flowmat(:,103,2)./flowmat(:,103,6)
% Sawmill residue not used, i.e. exported (negative value)
flowmat(:,104,:) = -1 * (1- parause(81)) * flowmat(:,102,:);
% Sawmill residue used as industrial fuel
flowmat(:,120,:) = parause(81) * flowmat(:,102,:) * parause(82);
% Sawmill residue used as a product in the local market
flowmat(:,121,:) = parause(81) * flowmat(:,102,:) * (1- parause(82));
%flowmat(:,102,2)./flowmat(:,102,6)
%*********************************************************************
% CALCULATION OF WOOD for PAPER PRODUCTION
%*********************************************************************
% Paper production and composition
flowmat(:,327,6) = parause(77) * 1000; % tonnes/a
% Nutrient and Energy content based on paper composition
flowmat(:,327,2:5) = flowmat(:,327,6) * parause2(144:147)' * (1 - parause2(148));
```

```
flowmat(:,327,7) = flowmat(:,327,6) * parause2(148);
flowmat(:,327,1) = flowmat(:,327,7) /1382000;

% Input of wood for paper production - a function of the pulp extraction
% process as described by parause(73)
flowmat(:,328,6) = flowmat(:,327,6)./ parause(73);
% Nutrient and Energy content based on wood composition only (no bark)
flowmat(:,328,2:5) = flowmat(:,328,6) * parause2(129:132)' * (1 - parause2(133));
flowmat(:,328,7) = flowmat(:,328,6) * parause2(133);
flowmat(:,328,1) = flowmat(:,328,7) /1382000;

%*********************************************************************
% CALCULATION WOOD PRODUCT REQUIREMENTS (population based)
%*********************************************************************
% Estimation of the annual internal use of wood products based on
% population (in metric tonnes).
flowmat(:,105,6) = statout(:,2).* (1000 * parause(78) * swcmtomt);
% Nutrient and Energy content based wood composition
flowmat(:,105,2:5) = flowmat(:,105,6) * parause2(129:132)';
% Total mass corrected for moisture content
flowmat(:,105,7) = flowmat(:,105,6)/(1 - parause2(133)) - flowmat(:,105,6);
flowmat(:,105,6) = flowmat(:,105,6)/(1 - parause2(133));
flowmat(:,105,1) = flowmat(:,105,7)/1382000;

% Estimation of the annual internal use of paper products based on
% population (in metric tonnes). Same proportion of softwood and hardwood as
% the one calculated from removals.
flowmat(:,122,6) = statout(:,2).* (1000 * parause(80) * swcmtomt);
% Nutrient and Energy content based paper composition
flowmat(:,122,2:5) = flowmat(:,122,6) * parause2(144:147)';
% Total mass corrected for moisture content
flowmat(:,122,7) = flowmat(:,122,6)/(1 - parause2(148)) - flowmat(:,122,6);
flowmat(:,122,6) = flowmat(:,122,6)/(1 - parause2(148));
flowmat(:,122,1) = flowmat(:,122,7)/1382000;

% Estimation of the annual internal use of firewood based on population
% (in metric tonnes).
flowmat(:,123,6) = statout(:,2).* (1000 * parause(79) * swcmtomt);
flowmat(:,123,2:5) = flowmat(:,123,6) * parause2(129:132)';
flowmat(:,123,7) = flowmat(:,123,6) * parause2(133);
flowmat(:,123,1) = flowmat(:,123,7)/1382000;

%*********************************************************************
% CALCULATION OF EXPORTS/IMPORTS OF WOOD
%*********************************************************************
% WOOD Production = REMOVALS - RESIDUE  (net production of green wood)
flowmat(:,100,:) = flowmat(:,97,:) - flowmat(:,98,:);

% Estimation of wood imports (includes paper and cardboard)
% ALL Consumption ****
% + Consumption by sawmills
% + Consumption by papermills
% + Internal wood consumption
% + Paper/cardboard consumption
Allcons = flowmat(:,101,:) + flowmat(:,328,:) + flowmat(:,105,:) + flowmat(:,122,:);

% All production ****
% - Internal wood production
% - Product from Sawmills
% - Used Residue from Sawmills
% - paper production
```

```
Allprod = flowmat(:,100,:) + flowmat(:,103,:) + flowmat(:,121,:) + flowmat(:,327,:);
%
%
flowmat(:,106,:) =  Allcons - Allprod;

% Estimation of total wood products exports
% sawmill residue + (-output)
flowmat(:,124,:) = flowmat(:,104,:) + flowmat(:,99,:);

%**********************************************************************
% ATMOSPHERIC PROCESSES
%**********************************************************************
% Calculations associated to nutrient atmospheric processes: deposition,
% volatilization, denitrification, and fixation
nut_depo

nut_fert

man_m = [parause2(113) parause2(118) parause2(123)]/100;

%*********************************************************************
% GENERATION/LOSSES OF MANURE nutrients (CATTLE/POULTRY/SWINE)
%*********************************************************************
manure

man_DM  = zeros(x,3); % years x [cattle poultry swine]
for a = 0:5:10
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
    b = a/5; % flowmat(:,309    flowmat(:,310   flowmat(:,311
    flowmat(:,309+b,6) = statout(:,5+b).* (statin(39+b) * parause(161+b));
    man_DM(:,b+1)      = flowmat(:,309+b,6).* (1 - man_m(b+1)); % tonne/y
    flowmat(:,309+b,7) = flowmat(:,309+b,6) - man_DM(:,b+1); % tonne/y
    flowmat(:,309+b,1) = flowmat(:,309+b,7)./1382000; % in Mgal/d
    flowmat(:,309+b,2:4) = man_DM(:,b+1) * parause2(109+a:111+a)'/100;
end
% manure lbs per head per day (crosschecking)
% for poultry
% (flowmat(:,310,6)./statout(:,6))/365 * 2.2
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% and A/C = 0.6783
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately
UAn     = zeros(3,6);    % [cattle poultry swine] x [C H S O N Ash]
Mener   = zeros(3,1); % [cattle poultry swine]
for a = 0:5:10
```

```
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
    b = a/5;
    Car = parause2(111+a);
    UAn(b+1,:) = [Car Car*0.1191 Car*0.0261 0 parause2(109+a) Car*0.6783];
    UAn(b+1,4) = 100 - (sum(UAn(b+1,1:3),2) + sum(UAn(b+1,5:6),2));

    % HHV in MJ/kg of dry mass
    Mener(b+1,1) = 0.3491 * UAn(b+1,1) + 1.1783 * UAn(b+1,2) ...
        + 0.1005 * UAn(b+1,3) - 0.1034 * UAn(b+1,4)...
        - 0.015 * UAn(b+1,5) - 0.0211 * UAn(b+1,6);
    % Energy in kWh/y
    flowmat(:,309+b,5) = man_DM(:,b+1).* Mener(b+1,1) * 1000 * 0.277778;
end
% TOTAL Manure Generated BEFORE handling and storage losses
% [cattle poultry swine]
flowmat(:,308,:) = flowmat(:,309,:) + flowmat(:,310,:) + flowmat(:,311,:);

%*********************************************************************
% STORAGE AND HANDLING OF MANURE CATTLE/POULTRY/SWINE
%*********************************************************************
% Manure handling/storage is known to be associated with the loss of about
% 35% in the case of poultry litter, 25% Anaerobic Pit, 60% Oxidation
% Ditch, and 80% Lagoon (the last three for cattle or swine) - NITROGEN.
% Source: Cecil Hammond, Retired Extension Engineer; Bill Segars, Extension
% Water Quality Specialist; & Charles Gould, Special Agent. "Land
% Application of Livestock and Poultry Manure",
% http://www.engr.uga.edu/service/extension/publications/c826-cd.html.
% Cattle[anaerobic pit] Poultry[litter pack] Swine[lagoon]
nLo = ones(x,1) * [0.25 0.35 0.80];

% We also asume that some carbon is lost due to handling and storage before
% final disposal or utilization of manure [Cattle Poultry Swine]
% Larney 2006   for cattle
% Nahm 2003     for poultry
% Velthof 2005  for Swine
cLo = ones(x,1) * [0.38 0.20 1];
% The approach to the equation also prevents NaN
for a = 1:x
    if man_DM(a,3) <= 0 % No swine manure production
        cLo(a,3) = 0;
```

```
    end
end
% After manure handling and storage some water content is lost due to
% evaporation. The FINAL water percentage (weight) is taken from different
% literature sources that included the use manure. As sorted in matrix:
% Cattle (Hao 2004, Eghball 2000)
% Poultry (kelleker 2002, Dias 2008, Mississippi Technology Alliance 2002)
% Swine (Fukumoto 2003)
% wLo was to be less than the values reported in man_m, fresh manure
% Check tryW = man_m - wLo has to be positive;
wLo = ones(x,1) * [0.60 0.40 0.70];

% Manure after storage and handling losses
% flowmat(:,179    flowmat(:,180   flowmat(:,181
flowmat(:,179:181,2) = flowmat(:,309:311,2).* (1- nLo);
flowmat(:,179:181,3) = flowmat(:,309:311,3);
flowmat(:,179:181,4) = flowmat(:,309:311,4).* (1- cLo);
flowmat(:,179:181,7) = flowmat(:,309:311,7).* (1- wLo);
flowmat(:,179:181,1) = flowmat(:,309:311,7)./ 1382000;
% Losses
nLosses = flowmat(:,309:311,2) - flowmat(:,179:181,2);
cLosses = flowmat(:,309:311,4) - flowmat(:,179:181,4);
wLosses = flowmat(:,309:311,7) - flowmat(:,179:181,7);
% Final manure mass after losses
flowmat(:,179:181,6) = flowmat(:,309:311,6) - (nLosses + cLosses + wLosses);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.1191, S/C = 0.0261,
% O/C = 0.9534, and A/C = 0.6783
UAn     = zeros(3,6);    % [cattle poultry swine] x [ultimate analysis]
Mener   = zeros(3,1); % [cattle poultry swine]
% Percentage of N and C in dry mass basis
PercN = flowmat(:,179:181,2)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
PercC = flowmat(:,179:181,4)./(flowmat(:,179:181,6) - flowmat(:,179:181,7)) * 100;
% Preventing NaN
PercN(isnan(PercN)) = 0;
PercC(isnan(PercC)) = 0;

man_DM2 = (flowmat(:,179:181,6) - flowmat(:,179:181,7));
for a = 1:3
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
    UAn(a,:) = [PercC(1,a) PercC(1,a)*0.1191 PercC(1,a)*0.0261 ...
```

```
        PercC(1,a)*0.9534 PercN(1,a) PercC(1,a)*0.6783];
    % HHV in MJ/kg of dry mass
    Mener(a,1) = 0.3491 * UAn(a,1) + 1.1783 * UAn(a,2) + 0.1005 * ...
        UAn(a,3) - 0.1034 * UAn(a,4)...
        - 0.015 * UAn(a,5) - 0.0211 * UAn(a,6);
    % Energy in kWh/y
    flowmat(:,179+a-1,5) = man_DM2(:,a).* Mener(a,1) * 1000 * 0.277778;
end
% Losses from manure storage and handling losses
flowmat(:,294,:) = -1 * (sum(flowmat(:,309:311,:),2) -  sum(flowmat(:,179:181,:),2));
% No energy content in emissions
EneLoss = flowmat(:,294,5); % Variable of lost energy
flowmat(:,294,5) = 0;

% TOTAL Manure Generated after handling and storage losses
% [cattle poultry swine] added
flowmat(:,182,:) = flowmat(:,179,:) + flowmat(:,180,:) + flowmat(:,181,:);

%*****************************************************************
% Recommended Nutrient FOR EACH LAND USE
%*****************************************************************
% Recommended Nutrient per land use (in tonne/km2)
% Matrix of nutrient content, rows [grass forestry crops] x columns [N P C]
% Pre-allocating
rec_fert = zeros(3);
bB = 92; % Initializing
for a = 1:3 % [grass forestry crops]
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
    for b = 1:3 % [N P C]
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
       rec_fert(a,b) = parause2(bB + b)./10;
    end
    bB = bB + b + 1;
end
%*****************************************************************
% Nutrient Content for each type of fertilizer IF SS PL CM SM CP
%*****************************************************************
% First we calculate the water content (moisture) of manure after storage
% & handling (estimated in manure.m) as a fraction w/w of total mass,
% nutrient content as w/w fraction of DM, and energy in kWh/tonne
fm_cont = zeros(x,tfe,nmat); % years x [IF SS PL CM SM CP] x species

% Data is loaded in a non-corrected way (i.e. Mgal/d, tonne/y, kWh/y)
% Column of zeros for IF
for a = 1:nmat
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
        flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
```

```
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
        fm_cont(:,:,a) = [zeros(x,1) flowmat(:,289,a) flowmat(:,180,a)...
            flowmat(:,179,a) flowmat(:,181,a) zeros(x,1)];
    end
    % Now data is corrected to reflect fraction of mass. For compost we use the
    % typical compost composition entered by user
    % Dry matter in tonne/y
    fm_drym =  fm_cont(:,:,6) - fm_cont(:,:,7);
    % We convert first column from Water content Mgal/d to w/w of total mass
    fm_cont(:,:,1) = fm_cont(:,:,7)./fm_cont(:,:,6);

    % Nutrient content corrected to w/w of dry matter
    for a = 2:4
        fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
        fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
        fm_cont(:,:,a) = fm_cont(:,:,a)./fm_drym;
    end
    % Energy content corrected to kWh per tonne (fresh matter)
    fm_cont(:,:,5) = fm_cont(:,:,5)./fm_cont(:,:,6);

    % fm_cont(:,:,7) and fm_cont(:,:,6) not used, thus this two terms are not a
    % fraction as from 1 to 5 are.
    % Data is added for IF (not calculated above) from user input
    % Assumed moisture 0.5%
    fm_cont(isnan(fm_cont)) = 0;
    fm_cont(:,1,1) = ones(x,1) * 0.005;
    fm_cont(:,1,modset(7)+1) = ones(x,1) * parause2(105 + modset(7) - 1)./100;

    % Data of ideal compost is added from user input
    fm_cont(:,6,1) = parause2(128)/100;
    fm_cont(:,6,2:4) = ones(x,1) * parause2(124:126)'/100;
    % energy calculated from ultimate analysis
    % Energy in kWh per tonne of total mass (wet basis)
    EneComp % fm_cont(:,6,5) calculated

    UAn     = zeros(x,6);    % percentage [C H S O N Ash]

    UAn(:,1) = fm_cont(:,6,4) * 100;
    UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
    UAn(:,5) = fm_cont(:,6,2) * 100;
    UAn(:,6) = UAn(:,1) * 1.4;
    UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

    %Energy in MJ/kg
    comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
            + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
            - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

    % Energy in kWh per tonne of total mass (wet basis)
    fm_cont(:,6,5) = comEner * 1000 * 0.277778 * (1 - parause2(128)/100);

    %*****************************************************************
    % NUTRIENT APPLICATION RATE
```

```
%*****************************************************************
% Calculates the amount of nutrient required (to comply with the minimum
% amount recomended for each fertilizer type and each lan use. Units in
% tonnes of fertilizer DRY MATTER per km2
%
% General equation: (REQUIRED = RECOMMENDED/CONTENT in FERT)
% rec_fert  = [grass forestry crops] x [N P C]
% fm_cont   = years x [IF SS PL CM SM CP] x species (w/w d.m.)
% req_fert  = [IF SS PL CM SM CP] x  [gr fo cr] x [years]
req_fert = zeros(tfe,3,x);

% First, fertilizers from organic sources IF SS PL CM SM CP are corrected
% by the corresponding plant available factor, one for N and one for P.
% P composting - no difference / N composting - reduced availability by 5
PAN = [1 ones(1,4)*parause(143) parause(143)/5];
PAP = [1 ones(1,5)*parause(144)];

% tonnes of DRY MATTER per km2 per year to satisfy limiting nutrient
% specified by user 'modset(7)' N or P in 'readme' of msainput.xls
for a = 1:x
    if modset(7) == 1
        FM_cont = fm_cont(a,:,modset(7)+1).* PAN;
    end
    req_fert(:,:,a) = (rec_fert(:,modset(7)) * FM_cont.^-1)';
end
% Preventing NaN
AreInf = (isinf(req_fert)==1);
req_fert(AreInf) = 0;

% Actual amount of fertilizer applied for each land use and type of
% fertilizer [IF SS PL CM SM CP]. Units tonnes/y of fertilizer Dry Matter (DM)
% Pre-allocating matrix space
fe_applied = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [grass forestry crops] x [years]
fe_mixarea0 = fe_mixarea;  % fe_mixarea created in fertmix.m
fe_mixarea0(1,:,:) = [];   % First row (non-fertilized areas) eliminated

% Actual fertilizer mass applied as tonne/y of dry matter
for a = 1:x % each year
    fe_applied(:,:,a) = fe_mixarea0(:,:,a).* req_fert(:,:,a); % tonnes/y
end
% Calculates the flow of all constituents in the fertilizer applied
% by type of fertilizer as calculated by the limiting nutrient.
% fm_cont = years x [IF SS PL CM SM CP] x species
for a = 1:x % year
    for b = 0:3 % [IF SS PL CM]
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
```

```
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
        % flowmat(:,135,:), flowmat(:,136,:), flowmat(:,137,:), flowmat(:,138,:)
        flowmat(a,135+b,1)   = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,1)/(1-fm_cont(a,b+1,1));
        flowmat(a,135+b,6)   = sum(fe_applied(b+1,:,a),2) + flowmat(a,135+b,1);
        flowmat(a,135+b,1)   = flowmat(a,135+b,1)./ 1382000; % converted to Mgal/d
        flowmat(a,135+b,2:4) = sum(fe_applied(b+1,:,a),2).* fm_cont(a,b+1,2:4);
        flowmat(a,135+b,5)   = flowmat(a,135+b,6).* fm_cont(a,b+1,5);
        flowmat(a,135+b,7)   = flowmat(a,135+b,1) * 1382000;
    end
end
% Swine Manure (SM) applied ...added last minute. flowmat(:,183
for a = 1:x % year
    flowmat(a,183,1)   = sum(fe_applied(5,:,a),2).* fm_cont(a,5,1)/(1-fm_cont(a,5,1));
    flowmat(a,183,6)   = sum(fe_applied(5,:,a),2) + flowmat(a,183,1);
    flowmat(a,183,1)   = flowmat(a,183,1)./ 1382000; % converted to Mgal/d
    flowmat(a,183,2:4) = sum(fe_applied(5,:,a),2).* fm_cont(a,5,2:4);
    flowmat(a,183,5)   = flowmat(a,183,6).* fm_cont(a,5,5);
    flowmat(a,183,7)   = flowmat(a,183,1) * 1382000;
end
% Compost (CP) applied ...added last minute. flowmat(:,316
for a = 1:x % year
    flowmat(a,316,1)   = sum(fe_applied(6,:,a),2).* fm_cont(a,6,1)/(1-fm_cont(a,6,1));
    flowmat(a,316,6)   = sum(fe_applied(6,:,a),2) + flowmat(a,316,1);
    flowmat(a,316,1)   = flowmat(a,316,1)./ 1382000; % converted to Mgal/d
    flowmat(a,316,2:4) = sum(fe_applied(6,:,a),2).* fm_cont(a,6,2:4);
    flowmat(a,316,5)   = flowmat(a,316,6).* fm_cont(a,6,5);
    flowmat(a,316,7)   = flowmat(a,316,1) * 1382000;
end
% verification of nutrient application and calculation of the required
% extra nutrient (non-limiting) that will be supplied as inorganic
% fertilizer, only if in deficit.
if modset(7) == 1;
    nlimN = 2; %Phosphorus is the non-limiting Nutrient
end
% nutrient fertilizer actually applied, tonne/y N or P
Actlimiting1 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] limiting
Actlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years] non-limiting
for a = 1:x % year
    for b = 1:3 % [grass forest crops]
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
       Actlimiting1(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,modset(7)+1)';
       Actlimiting0(:,b,a) = fe_applied(:,b,a).* fm_cont(a,:,nlimN + 1)';
    end
end
%*************************************************************************
% Plant minimum requirement of the non-limiting nutrient, tonne/y N or P
Reqlimiting0 = zeros(tfe,3,x); % [IF SS PL CM SM CP] x [g f c] x [years]
for a = 1:x % year
    for b = 1:5 % [IF SS PL CM SM]
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
       Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
```

```
        Reqlimiting0(b,:,a) = fe_mixarea0(b,:,a).* rec_fert(:,nlimN)';
    end
end
% We finally identify those instance where there is deficit of non-limiting
% nutrient (positive values) and those instances where there is excess will
% be ignored. Deficit occurs because the heterogenity of the nutrient
% sources: manure, etc
Deficit = Reqlimiting0 - Actlimiting0;
Deficit = (abs(Deficit) + Deficit)/2; % Eliminate negative values
% Imported non-limiting nutrient in inorg form to meet plant requirement
Implimiting0 = zeros(x,4,nmat); % 4 for Total, grass, forest, and crops
% non-limiting nutrient mass tonne/y
Implimiting0(:,1,nlimN + 1) = sum(sum(Deficit,1),2);
for a = 1:x
    Implimiting0(a,2:4,nlimN + 1) = sum(Deficit(:,:,a),1);
end
% Fertlizer dry mass (non-limiting)
NLdrym = Implimiting0(:,:,nlimN + 1)./(parause2(105 + modset(7))./100);
% moisture mass tonne/y
for a = 1:4
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
Implimiting0(:,a,7) = NLdrym(:,a).* fm_cont(:,1,1)./(1 - fm_cont(:,1,1));
end
% Total mass of fertilizer
Implimiting0(:,:,6) = Implimiting0(:,:,7) + NLdrym;
% moisture volume Mgal/d
Implimiting0(:,:,1) = Implimiting0(:,:,7)./ 1382000;

% Inorganic Fertlizer application is corrected with the non-limiting
% nutrient that is supplemented
flowmat(:,135,:) = flowmat(:,135,:) + Implimiting0(:,1,:);

% Total fertilizer and nutrient applied
flowmat(:,139,:) = sum(flowmat(:,135:138,:),2) + flowmat(:,183,:) + flowmat(:,316,:);

%**************************************************************************
%
% APPLICATION PER LAND USE. Added August 20010
%
%
%**************************************************************************
%
% flowmat(:,322,:), flowmat(:,323,:), flowmat(:,324,:)
% Calculates the flow of fertilizer applied by type of land use.
% fm_cont = years x [IF SS PL CM SM CP] x species
% fe_applied = [IF SS PL CM SM CP] x [grass forestry crops] x [years]
for a = 1:x % year
    for b = 0:2 % [grass forestry crops]
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
```

```
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
        flowmat(a,322+b,7) = sum(fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))');
        flowmat(a,322+b,6) = sum(fe_applied(:,b+1,a)) + flowmat(a,322+b,7);
        % For calculating energy content we estimate the fresh matter mass
        % in tonne/y as the sum of the previous two terms
        FreshMatter = fe_applied(:,b+1,a).* (fm_cont(a,:,1)./(1-fm_cont(a,:,1)))'...
            + fe_applied(:,b+1,a);
        flowmat(a,322+b,1) = flowmat(a,322+b,7)./ 1382000; % converted to Mgal/d
        flowmat(a,322+b,2)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,2)');
        flowmat(a,322+b,3)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,3)');
        flowmat(a,322+b,4)   = sum(fe_applied(:,b+1,a).* fm_cont(a,:,4)');
        flowmat(a,322+b,5)   = sum(FreshMatter.* fm_cont(a,:,5)');
    end
end
% The amounts are now corrected for the supplement fertilizer added to
% supply the deficiency calculated for the non-limiting nutrient (N or P)
flowmat(:,322,:) = flowmat(:,322,:) + Implimiting0(:,2,:);
flowmat(:,323,:) = flowmat(:,323,:) + Implimiting0(:,3,:);
flowmat(:,324,:) = flowmat(:,324,:) + Implimiting0(:,4,:);

%*****************************************************************
% MANURE distribution
%*****************************************************************
% Calculates de amount of manure sent to waste management and the manure
% that has to be imported from outside the system to satisfy the
% fertilizer demand as calculated in nut_fert.m, i.e flowmat(138),
% flowmat(137), and flowmat(183)
% Pre-allocating matrix space [ year CATTLE/POULTRY/SWINE species]
pHi = zeros(x,3,nmat);
% CONSUMED for fertilization - PRODUCED (Consumed calculated in nut_fert.m)
pHi(:,1,:) = flowmat(:,138,:) - flowmat(:,179,:);
pHi(:,2,:) = flowmat(:,137,:) - flowmat(:,180,:);
pHi(:,3,:) = flowmat(:,183,:) - flowmat(:,181,:);

for a = 1:x % year
    for b = 0:2 % CATTLE/POULTRY/SWINE
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        if pHi(a,b+1,modset(7)+1) > 0 % Based on limiting nutrient
        else
            % Manure sent to waste management
            % flowmat(:,188 flowmat(:,189 flowmat(:,190
```

```
            flowmat(a,188+b,:) = -1 * pHi(a,b+1,:);
        end
    end
end

%Total Manure utilized for soil fertilization
flowmat(:,252,:) = flowmat(:,138,:) + flowmat(:,137,:) + flowmat(:,183,:);
%Total Manure sent to waste management
flowmat(:,191,:) = flowmat(:,188,:) + flowmat(:,189,:) + flowmat(:,190,:);
%Total Manure imported for fertilization
flowmat(:,187,:) = flowmat(:,184,:) + flowmat(:,185,:) + flowmat(:,186,:);

%************************************************************************
% SEWAGE SLUDGE (TREATED) remaining amount after use as fertilizer
%************************************************************************
% Treated Sludge 'flowmat(:,289,:)' used straight as a soil fertilizer
% 'flowmat(:,136,:)'.
% Remaining Sludge for other waste management options in wastems.m
% Treated sludge minus sludge used for land application
remainTS = flowmat(:,289,:) - flowmat(:,136,:);

% Check that TS is enough at least for soil application, with respect to
% limiting nutrient, specified in 'modset'
if isreal(sqrt(remainTS(:,:,modset(7)+1))) == false

%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT DEPOSITION
%************************************************************************
% We assume that nutrient deposition is
% already considered in nutrient leaching and volatilization factors. In
% this sense, wet and dry deposition is considered as a background
% addition of nutrient
% Wet deposition (without the water component)
flowmat(:,132,2:4) = flowmat(:,81,2:4);
flowmat(:,132,6) = sum(flowmat(:,81,2:4),3);

% Dry deposition (water component is equal to zero and total mass flow is
% not calculated). Deposition rates in kg/ha.yr converted to tonne/y
dry_npc = parause2(45:47) * (land * 100) /1000; % 'land' converted to ha
flowmat(:,133,:) = ones(1,x)' * [0  dry_npc(1) dry_npc(2) dry_npc(3) 0 sum(dry_npc) 0];

% Total deposition
 flowmat(:,134,:) = flowmat(:,132,:) + flowmat(:,133,:);

%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Fixation
%************************************************************************
% 'Areas' matrix calculated in 'fertmix.m'
for a = 1:x % year
    flowmat(a,145,2) = sum(parause(124:126)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for Nitrogen Denitrification
%************************************************************************
% 'Areas' matrix calculated in 'fertmix'
for a = 1:x % year flowmat(:,146,2)
    flowmat(a,146,2) = -1 * sum(parause(127:129)'.* Areas(a,:)*100/1000);
end
%************************************************************************
% NUTRIENT CALCULATIONS for estimating NUTRIENT LEACHING
%************************************************************************
```

```
% Fertilizer applied by type and nutrient (N P C) in tonne/y
% Pre-allocating matrix space
nut_applied = zeros(5,3,x);
for a = 1:x % year
    for b = 1:4 % [IF SS PL CM]
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
        nut_applied(b,:,a) = flowmat(a,134+b,2:4);
    end
    % Swine Manure (SM)..last minute addition
    nut_applied(5,:,a) = flowmat(a,183,2:4);
end
leaching

floorR = parause(212:214); % [grass forestry crops]
% Assumed base drainage value, for leaching variation with drainage flow
AbaseD = 250; % mm/year

% Available water for percolation in all pervious area (non-irrigated)
% due to precipitation - evapotranspiration
CORRepa = perprecip(:,1,1) + flowmat(:,79,1) + surw_evap(:,:,1);
% Converted Mgal/d to m3/y
CORRepa = CORRepa * 3785.41178 * 365;
% Converted mm/y by dividing by all pervious area
CORRepa = (CORRepa./(statout(:,27) * 1000000)) * 1000;
% Matrix reformated
Drain = ones(tfe+1,3,x);
for a = 1:x
    Drain(:,:,a) = Drain(:,:,a) * CORRepa(a) * parause(59);
end
% complete
% Irrigated areas have more water available for infiltration but
% since irrigated areas are just a small portion of the whole area we are
% ignoring that for now. CORRepa includes the irrigation term but is
% aggregated to the rest of infiltrated water.
% Nutrient applied per unit of area (nitrogen and phosphorus) tonne/km2
% [NF IF SS PL CM SM CP] x [grass forestry crops] x [years] x [N P]
Nut_rate = zeros(tfe+1,3,x,2);
if modset(7) == 1
    Nut_rate(2:tfe+1,:,:,1) = Actlimiting1./ fe_mixarea0;
    Nut_rate(2:tfe+1,:,:,2) = (Deficit + Actlimiting0)./ fe_mixarea0;
end
% NaN eliminated
Nut_rate(isnan(Nut_rate)) = 0;
% Converted to kg per hectare
Nut_rate = Nut_rate * 1000/100;

% $%$$$$$$$$$$$$$$$$$$$$$$$$$
% Nitrogen leaching
% Leaching due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Nleaching = (Nut_rate(:,:,:,1) * parause(141)).* (Drain/AbaseD);
% Leaching in tonne/y
Nleaching = Nleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
```

```
    Nback(1,:,a) = parause2(262) * parause(141) * floorR; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nleaching(1,:,:) = Nleaching(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nleaching = sum(sum(Nleaching,1),2);
for a = 1:x;
    flowmat(a,83,2) = -Nleaching(:,:,a);
end
% Phosphorus leaching
% Leaching due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by drainage flow
% Leaching in kg/ha.y
Pleaching = (Nut_rate(:,:,:,2) * parause(142)).* (Drain/AbaseD);
% Leaching in tonne/y
Pleaching = Pleaching.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = parause2(263) * parause(142) * floorR; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Pleaching(1,:,:) = Pleaching(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Pleaching = sum(sum(Pleaching,1),2);
for a = 1:x;
    flowmat(a,83,3) = -Pleaching(:,:,a);
end
% Carbon leaching
% Fertilization is considered to have no effect on carbon leaching. But
% soil organic matter is assumed to have 100% responsability
% Background value calculated (for all areas) tonne/km2
Cback = ones(size(fe_mixarea));
for a = 1:x
    for b = 1:6
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
        Cback(b,:,a) = parause2(264) * parause(140) * floorR; % [grass forestry crops]
    end
end
Cback = Cback.* fe_mixarea.* (Drain/AbaseD);

% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,83,4) = -Cback(:,:,a);
end
% Concentration check [N P C]
LeachConc = zeros(x,3);
for a = 2:4
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);
    LeachConc(:,a - 1) = flowmat(:,83,a)./ (flowmat(:,83,1) * 1.382);

%*********************************************************************
```

```
% NUTRIENT CALCULATIONS for estimating NUTRIENT SURFACE RUNOFF
%*********************************************************************
% Calculates runoff flow per land use (PA only) and its nutrient content
runoff_pa

Pratio = ones(tfe+1,3,x); %[nf IF SS PL CM SM CP] x [gr fors crops] x years
for a = 1:x
    Pratio(:,:,a) = Pratio(:,:,a) * inout(a,1)/statin(27);
end
% Nitrogen runoff
% Runoff due to nitrogen fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Nrunoff = (Nut_rate(:,:,:,1) * parause(210)).* Pratio;
% Leaching in tonne/y
Nrunoff = Nrunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Nback = zeros(size(fe_mixarea));
for a = 1:x
    Nback(1,:,a) = parause2(21:4:29)' * 100/1000; % [grass forestry crops]
end
Nback = Nback .* fe_mixarea;
Nrunoff(1,:,:) = Nrunoff(1,:,:) + Nback(1,:,:);

% Aggregated matrix
Nrunoff = sum(sum(Nrunoff,1),2);
for a = 1:x;
    flowmat(a,77,2) = flowmat(a,77,2) + Nrunoff(:,:,a);
end
% Phosphorus runoff
% Runoff due to phosphorus fertilization plus background value
% (undisturbed forest value) and corrected by precipitation
% Runoff in kg/ha.y
Prunoff = Nut_rate(:,:,:,2).* parause(211).* Pratio;
% Leaching in tonne/y
Prunoff = Prunoff.* fe_mixarea * 100/1000;

% Background value calculated (only to non-fertilized areas) tonne/km2
Pback = zeros(size(fe_mixarea));
for a = 1:x
    Pback(1,:,a) = [parause2(22:4:30)]' * 100/1000; % [grass forestry crops]
end
Pback = Pback .* fe_mixarea;
Prunoff(1,:,:) = Prunoff(1,:,:) + Pback(1,:,:);

% Aggregated matrix
Prunoff = sum(sum(Prunoff,1),2);
for a = 1:x;
    flowmat(a,77,3) = flowmat(a,77,3) + Prunoff(:,:,a);
end
% Carbon runoff
% background value is calculated based on the O horizon carbon content,
% similar to leaching, and corrected by precipitation
Cback = zeros(size(fe_mixarea));
for a = 1:x
    Cback(1,:,a) = parause2(264) * parause(139) * floorR; % [grass forestry crops]
end
Cback = Cback .* fe_mixarea.* Pratio;
```

```
% Aggregated matrix
Cback = sum(sum(Cback,1),2);
for a = 1:x;
    flowmat(a,77,4) = flowmat(a,77,4) + Cback(:,:,a);
end
% Concentration check
RunoffConc = zeros(x,3);
for a = 2:4
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
    RunoffConc(:,a - 1) = flowmat(:,77,a)./ (flowmat(:,77,1) * 1.382);
end
%************************************************************************
%****************** (originally calculated in waters) ********************
% Total runoff generated from IA + PA
flowmat(:,84,2:4) = flowmat(:,76,2:4) + flowmat(:,77,2:4);
% Total runoff that reaches the river (IA + PA - Sewer Inflow)
flowmat(:,295,2:4) = -1 * (flowmat(:,84,2:4) - sinflow(:,1,2:4));
%************************************************************************

 %************************************************************************
% ESTIMATION OF SURFACE WATERSHED DISCHARGE (RIVER FLOW)
%************************************************************************
% Stream flow =
% + 295 Total surface run-off to stream
% +  95 Baseflow contribution
% +  93 Total water returned to surface sources
% -   2 Total water withdrawals
% - 286 Water surface storage (lake)
% - 'surw_evap' Evaporation from surface storage
% The last two terms are assumed to be the amount of water released
% from the surface storage structure since '2' already accounts for
% withdrawals from reservoirs and river.
flowmat(:,296,:) = -flowmat(:,295,:) + flowmat(:,95,:) ...
    - flowmat(:,93,:) - flowmat(:,2,:) - flowmat(:,286,:) - surw_evap;

%flowmat(:,296,1) * 22.8245; % converting to Mgd from m3/s
% Calculation of RMSE (water balance only)
% [water N P C] x [rmse rrmse]
% check: adjust parameters to calibrate to known river flow, min rmseR(1,2)
% things to manipulate:
% IBT, parameter59, parameter64,
rmseR = zeros(4,2);
rmseR(1,1) = sqrt((sum((flowmat(:,296,1) - inout(:,2)).^2))/x);
rmseR(1,2) = rmseR(1,1) *100 / mean(inout(:,2));

% For calibration purposes. UCW.
% USGS sample site. CHATTAHOOCHEE RIVER AT GA 280, NEAR ATLANTA, GA.
%
% check concentration TN in mg/L - 3.00 (1.50-4.50)
% Mgald = flowmat(:,296,1)./(1000000 * 0.0037854 / (24 * 3600));
% flowmat(:,296,2)./(Mgald  * 1.382)
% check concentration TP in mg/L - 0.50 (0.20-0.90)
% flowmat(:,296,3)./(Mgald  * 1.382)
% check concentration TOC in mg/L - 5.00 (2.00-9.00)
% flowmat(:,296,4)./(Mgald  * 1.382)
%************************************************************************
% CONVERSIONS
%************************************************************************
% Water volume to mass
% Converts Mgal/d to tonne/y (for almost 100% water flows)
```

```
flowmat(:,1:96,6) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,6) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,6) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,6) = flowmat(:,320,1) * 1382000;
flowmat(:,330,6) = flowmat(:,330,1) * 1382000;

% Converts Mgal/d to tonne/y (for almost 100% water flows)
flowmat(:,1:96,7) = flowmat(:,1:96,1) * 1382000;
flowmat(:,284:286,7) = flowmat(:,284:286,1) * 1382000;
flowmat(:,295:296,7) = flowmat(:,295:296,1) * 1382000;
flowmat(:,320,7) = flowmat(:,320,1) * 1382000;
flowmat(:,330,7) = flowmat(:,330,1) * 1382000;

% Correction of the energy content of water flows. For liquid flows it is
% calculated as the thermal enegy with respect to the average ambient
% temperature, while for vapor flows it is calculated as the vaporization
% energy (from water latent heat).
waterEner

En1 = watEmat(watEmat(:,1) == 1,3);
for a = 1:x
    % kWh per year
    flowmat(a,En1,5) = flowmat(a,En1,7).* watEmat(watEmat(:,1) == 1,2)' * 1.1622;
end
% Water flows that are vapor (vaporization energy)
En2 = watEmat(watEmat(:,1) == 2,3);
for a = 1:x
    % kWh per year
    flowmat(a,En2,5) = flowmat(a,En2,7).* wl * 1.1622;
end
% Compounded flows are calculated
% Consumptive use of Power Generation - Evap
flowmat(:,60,5) = -sum(flowmat(:,50:56,5),2);
% Consumptive use of Fuel Production - Evap
flowmat(:,71,5) = sum(flowmat(:,65:67,5),2);
% Total surface run-off generated (IA + PA)
flowmat(:,84,5) = sum(flowmat(:,76:77,5),2);
% Evapotranspiration (IA + PA)
flowmat(:,85,5) = sum(flowmat(:,78:79,5),2);
% Other evaporation (consumptive IN + PG + FP)
flowmat(:,86,5) = sum([flowmat(:,30,5) flowmat(:,60,5) flowmat(:,71,5)],2);
% Total water evaporation
flowmat(:,87,5) = sum(flowmat(:,85:86,5),2);
% Other water returns to surface sources (IN, PG, and FP)
flowmat(:,92,5) = sum([flowmat(:,29,5) flowmat(:,61,5) flowmat(:,72,5)],2);
% Total water returned to surface sources
flowmat(:,93,5) = -sum(flowmat(:,91:92,5),2);
% Surface runoff that reaches the river (output)
% flowmat(:,84,:) - sinflow
fact295 = (flowmat(:,84,1) - sinflow(:,:,1))./flowmat(:,84,1);
flowmat(:,295,5) = -flowmat(:,84,5).* fact295;

%*********************************************************************
% NUTRIENT CALCULATIONS for estimating NITROGEN VOLATILIZATION
%*********************************************************************
% Volatilization for each fertilizer type
% Pre-allocating matrix space
vol_fert = zeros(x,5); %[years] x [IF SS PL CM SM]
for a = 1:x % year
    vol_fert(a,:) = nut_applied(:,1,a).* [parause(133:136)' parause(136)]';
```

```
end
flowmat(:,140,2) = -1 * sum(vol_fert,2);  % tonne/y
flowmat(:,140,6) = flowmat(:,140,2); % tonne/y
% Calculations associated to respiration (human, animal, and vegetation)
nut_resp

Pnumber = [statout(:,2) statout(:,5) statout(:,6) statout(:,7)];
% Calculation of C releases per capita/head (kg C/year)
Mc = [3.32 * statin(53)^0.79 (3.32 * statin(39:41).^0.79)'];
% Calculation for livestock per capita/head (kg C/year)
Ec = [3.10 * statin(53)^0.63 (3.10 * statin(39:41).^0.63)'];

% Total releases kg C/year per capita/head [humans cattle poultry swine]
Creleases = Mc;% + Ec;
% Ec not considered as the carbon emission associated to decomopsition of
% manure and human waste is calculated in other processes. Avoiding double
% counting
% Totals in tonne C per year, flowmat(:,147
for a = 1:x
    flowmat(a,147,4) = -1 * sum(Pnumber(a,:).* Creleases);

% MODULE Food sector
foods;

nut_live

BW = [statin(39) statin(40) statin(41)];

% Food rate consumption Cattle  as DE, kcal/head.d
% Milk cattle
% Milk production factor kg/d (assumed 3.5% fat)
FCM = 0.4 * (statin(44) / 365) + 15 * (statin(44) / 365) * 0.035;
DW = 0; % weight gain assumed kg/d
% First we calculate TDN kg/d
f_mil = 0.035 * BW(1)^0.75 + 0.33 * FCM + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Beef cattle
f_bee = 0.035 * BW(1)^0.75 + BW(1)^0.75 * (0.05603 * DW...
    + 0.01265 * DW^2)/2.3;
% Conversion to DE
f_mil = f_mil * 4400;
f_bee = f_bee * 4400;

% Food rate consumption Poultry as DE, kcal/head.d
% Egg weght, grams per day, using "Predicting feed intake of food-producing
% animals" By National Research Council, US - Subcommittee on Feed Intake
% Table 4-1 Page 46
EggW = (23.882 * BW(2) + 15.639) * (statin(45)/365);
DW = 0.1; % weight gain assumed g/d
% Digestible energy DE, Kcal/head.day
f_lay = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 0.20 * EggW + 147;
DW = 0.5; % weight gain assumed g/d
f_bro = (1/0.82) * 0.532 * (BW(2) * 1000)^0.75 + 14.5 * DW + 147;

% Food rate consumption Pigs (swine) as DE, kcal/head.d. excel file.
if BW(3) < 20
else % from swine > 20 kg
    f_swi = 13162 * (1 - exp(-0.01768 * BW(3)));
end
% Adjustements due to environmental and living conditions
Tzero = 26 - 0.0614 * BW(3);
```

```
adj1 = 1 + 0.0165 * (Tzero - statin(35))/100;
if BW(3) > 25
    adj2 = 1 + (0.2142 * BW(3) - 0.00133 * BW(3)^2 - 4.42)/100;
end
f_swi = f_swi * adj1 * adj2;

% Digestible energy requirements Mcal/y
DE = zeros(x,3); % [cattle poultry swine]
DE(:,1) = statout(:,5).* (f_mil * statin(42) + f_bee * (1-statin(42))) * 365;
DE(:,2) = statout(:,6).* (f_lay * statin(43) + f_bro * (1-statin(43))) * 365;
DE(:,3) = statout(:,7).* f_swi * 365;

% Assumed moisture content for each feed
m_feed = [0.12 0.12 0.12];  % [Cattle Poultry Swine]
% Dry mass of feed in tonne/y
DM_all = zeros(x,3); % [cattle poultry swine]
for a  = 0:2
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
    DM_all(:,a + 1) = DE(:,a + 1)./parause2(205 + (a * 5))';
end
% Consistency check. Feed intake kg per head daily.
% cattle    DM_all(:,1)./(statout(:,5) * 365)   around 5-10 kg
% poultry   DM_all(:,2)./(statout(:,6) * 365)   around 0.05 - 0.09 kg
% swine     DM_all(:,3)./(statout(:,7) * 365)   around 1-4 kg
for b = 0:5:10 % [cattle poultry swine]
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
    a = b/5 + 1;
    flowmat(:,173+a-1,2) = DM_all(:,a).* (parause2(201+b)/100)/6.25;
    flowmat(:,173+a-1,3) = DM_all(:,a).* (parause2(204+b)/100);
    flowmat(:,173+a-1,4) = DM_all(:,a).* ((parause2(201+b)/100)/1.98 + ...
        (parause2(202+b)/100)/1.357 + (parause2(203+b)/100)/2.50);
    flowmat(:,173+a-1,5) = DM_all(:,a).*1000 * parause2(205+b) * 0.0011622232;
    flowmat(:,173+a-1,6) = DM_all(:,a)./(1 - m_feed(a));
    flowmat(:,173+a-1,7) = DM_all(:,a).* m_feed(a)/(1 - m_feed(a));
    flowmat(:,173+a-1,1) = flowmat(:,173+a-1,7)./ 1382000;
end
% We correct the energy calculated by digestible energy (DE) and converted
% to gross energy which is assumed to be 75%
flowmat(:,173:175,5) = flowmat(:,173:175,5)./0.75;

% Protein and phosphorus content is corrected to reach minimum requirements
% Based on the National Research Council a minimum of 15% CP and 0.6% P are
```

```
% required for Poultry (NRC 1961 in WPS Journal).
% All requirements follow National Research Council recommendations.
% Equations were derived from, see 'livestock and farm inventory.xls':
% Nutrient Requirements of Swine, NRC 2000
% Minimum requirements for all livestock population, tonne/y
minN = zeros(x,3,2); % [years] [cattle poultry swine] [ N and P]
% DE in Mcal/y and rate in kg/kcal
% Cattle (beef and dairy)
minN(:,1,1) = 7.84688E-06 * DE(:,1);
minN(:,1,2) = 1.19415E-06 * DE(:,1);
% Poultry
minN(:,2,1) = DM_all(:,2) * 0.15/6.25;
minN(:,2,2) = DM_all(:,2) * 0.006;
% Swine
if BW(3) < 20
else
    Pnit = -1.178E-04 * BW(3) + 3.276E-02;
    Ppho = -1.535E-05 * BW(3) + 5.524E-03;
end
minN(:,3,1) = DM_all(:,3) * Pnit;
minN(:,3,2) = DM_all(:,3) * Ppho;


% Nutrient deficit, N and P
defN = minN - flowmat(:,173:175,2:3);
defN = (abs(defN) + defN)/2; % Eliminate negative values
flowmat(:,307,2:3) = sum(defN,2);

% Regular feed plus nutrient supplements, only N and P
flowmat(:,173:175,2:3) = flowmat(:,173:175,2:3) + defN;
flowmat(:,173:175,6) = flowmat(:,173:175,6) + sum(defN,3);

% Feed consumed by pets
pet_food = zeros(x,1,nmat);
% Moisture content assumed same as for cattle feed
DM_pet = parause(148) * statout(:,2) * (1 - m_feed(1));
pet_food(:,:,2) = DM_pet.* (parause2(266)/100)/6.25;
pet_food(:,:,3) = DM_pet.* (parause2(269)/100);
pet_food(:,:,4) = DM_pet.* ((parause2(266)/100)/1.98 + ...
        (parause2(267)/100)/1.357 + (parause2(268)/100)/2.50);
pet_food(:,:,5) = DM_pet.*1000 * parause2(270) * 0.0011622232;
pet_food(:,:,6) = DM_pet./(1 - m_feed(1));
pet_food(:,:,7) = DM_pet.* m_feed(1)/(1 - m_feed(1));
pet_food(:,:,1) = pet_food(:,:,7)./ 1382000;

% All animal feed consumed (including pets)
flowmat(:,176,:) = flowmat(:,173,:) + flowmat(:,174,:) + ...
    flowmat(:,175,:) + pet_food;

% check that livestock is not excreting more than what is eating
% f=0    f=1    f=2 [cattle poultry swine]
% excreting/feed
% flowmat(:,309+f,2)./flowmat(:,173+f,2)
% flowmat(:,309+f,3)./flowmat(:,173+f,3)
if any(flowmat(:,308,2:4) > flowmat(:,173,2:4) + flowmat(:,174,2:4) + flowmat(:,175,2:4))
    end
else
    emat(19) = 0;
end
%*****************************************************************
% PRODUCTS FROM LIVESTOCK (INTERNAL PRODUCTS)
```

```
%*******************************************************************
% FISH  ************************************************************
flowmat(:,161,:) = 0;     % No significant fish production in the Upper
                           % Chattahoochee Watershed
% MEAT  ************************************************************
% Produced [Cattle Poultry Pigs] in tonnes/y calculated from Body Weight and
% from livestock inventory at the begining of the year
P_meat = zeros(x,3,nmat);
% [years] x [Cattle Poultry Pigs]
for b = 0:2
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
    P_meat(:,b + 1,6) = livestock(2:x+1,b+1) .* parause(158+b)* statin(39+b);
end

% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
    a = (b/4)+1;
    P_meat(:,a,2) = P_meat(:,a,6).* parause2(189+b);
    P_meat(:,a,3) = P_meat(:,a,6).* parause2(190+b);
    P_meat(:,a,4) = P_meat(:,a,6).* parause2(191+b);
    P_meat(:,a,5) = P_meat(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    P_meat(:,a,7) = P_meat(:,a,6).* m_meat(a);
    P_meat(:,a,1) = P_meat(:,a,7)./ 1382000;
end
% Total meat produced
flowmat(:,160,:) = sum(P_meat,2);

% Total animal (meat) mass accumulated due to increase or decrease of
% livestock (live animal): [year] x [Cattle Poultry Pigs] x [specie]
L_acc = zeros(x,3,nmat);
% Average livestock growth in tonne/y, based on growth of thousand
% heads: 'liveg' calculated in 'livepop.m'
for b = 0:2
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
    L_acc(:,b + 1,6) = liveg(:,b + 1) .* statin(39 + b);
end
% [year] x [Cattle Poultry Pigs] x [specie]
for b = 0:4:8
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
```

```
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
    a = (b/4)+1;
    L_acc(:,a,2) = L_acc(:,a,6).* parause2(189+b);
    L_acc(:,a,3) = L_acc(:,a,6).* parause2(190+b);
    L_acc(:,a,4) = L_acc(:,a,6).* parause2(191+b);
    L_acc(:,a,5) = L_acc(:,a,6).* 1000 * parause2(192+b) * 0.0011622232;
    L_acc(:,a,7) = L_acc(:,a,6).* m_meat(a);
    L_acc(:,a,1) = L_acc(:,a,7)./ 1382000;
end
% Total livestock accumulation
flowmat(:,306,:) = sum(L_acc,2);

% DAIRY  ************************************************************
% Egg refuse portion (shell) is 10.5% (The Chemical Composition of American
% Food Materials (1896) Professor W.O. Atwater). For the total mass
% will add again the refure mass.
re_mass = 0.105;
egg_size = (23.882 * statin(40) + 15.639)*(1 - re_mass); % g of edible part per unit

% Total dairy mass, milk and eggs in tonne per year
mil_mass = statout(:,5).* statin(42)* statin(44);
egg_mass = statout(:,6).* statin(43)* statin(45) * egg_size/1000;

d_mass = [egg_mass mil_mass];
% Matrix for diary production
P_dairy = zeros(x,2,nmat); % [years] x [eggs milk] x [specie]

% For correlation between carbohydrates, protein, fat, nitrogen, carbon,
% see notes page 53. Correlations derived from molecular formulas.
for b = 0:5:5
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
    a = b/5+1;
    P_dairy(:,a,2) = d_mass(:,a).* (parause2(216+b)/100)/6.25;
    P_dairy(:,a,3) = d_mass(:,a).* (parause2(219+b)/100);
    P_dairy(:,a,4) = d_mass(:,a).* ((parause2(216+b)/100)/1.98 + (parause2(217+b)/100)/1.357 +
(parause2(218+b)/100)/2.50);
    P_dairy(:,a,5) = d_mass(:,a).*1000 * parause2(220+b) * 0.0011622232;
    P_dairy(:,a,6) = d_mass(:,a);
    P_dairy(:,a,7) = d_mass(:,a).* m_dai(a);    % tonnes/y
    P_dairy(:,a,1) = P_dairy(:,a,7)./ 1382000;  % Mgal/d
end
% We correct the total mass of eggs by including its refuse fraction
P_dairy(:,1,6) = P_dairy(:,1,6)/(1 - re_mass);

% Total dairy produced
flowmat(:,162,:) = sum(P_dairy,2);
```

```
% Nutrient consistency check
% Feed should be larger that (meat + diary + manure'before losses')
if any(flowmat(:,160,2:4) + flowmat(:,162,2:4) + flowmat(:,308,2:4) > flowmat(:,176,2:4))
    end
else
    emat(20) = 0;

nut_food

crop_a = statout(:,11)./statout(1,11);

% FRUITS  ************************************************************
% Area array = [area values in km2 for as many as fruits considered]
% Orchards in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) grapes 46% peach 16% berries 16% pecan 14% apples 7%]
% SPECIFIC FOR UCW (not year specific though)
% [grapes peach berries pecan apples]
% Total area (426 acres) calculated in 'UC nutrient.xls' for year 2000
%fru_area = 426 * 0.0040468564 * [0.46 0.17 0.16 0.14 0.07]; %km2
% Fruits are lump
fru_area = statout(1,11) * fr_mix(:,2)'/100; % km2
% Yield + uncertainty in tonne/km2.a
fru_yield = fr_mix(:,3)' * parause(198);

% Production after yield losses for first year
fru_prod = sum((fru_area.* fru_yield) * (1 - parause(138))); % tonne/y
% Calculation of fruit production
flowmat(:,157,6) = fru_prod * crop_a;
% Multiplier
mult = [m_food(1) parause2(149:151)' parause2(152)*1.1622232];
% Composition of fruits
flowmat(:,157,1:5) = flowmat(:,157,6) * mult;   % tonnes/y and kWh/y
flowmat(:,157,7) = flowmat(:,157,1);           % tonnes/y
flowmat(:,157,1) = flowmat(:,157,7)./ 1382000;  % converted to Mgal/d

% CEREAL  ************************************************************
% [cornforgrain wheatforgrain]
% Area array = [area values in km2 for as many as cereals considered]
% Total area (860 acres) calculated in 'UC nutrient.xls' for year 2000
%cer_area = 860 * 0.0040468564 * [0.84 0.16]; %km2 UCW
cer_area = statout(1,11) * ce_mix(:,2)'/100; % km

% Cereal Yield. Original data in bushels per acre but converted to CWT
% by using 60 lbs/bushel for wheat and 56 lbs/bushel for corn. CWT is
% exactly 100 pounds. See 'Crops Inventory 2002.xls' UCW
%cer_yield = [45 25]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
cer_yield = ce_mix(:,3)' * parause(199); %tonne/km2.y

% Production after yield losses for year 2000
cer_prod = sum((cer_area.* cer_yield) * (1 - parause(138))); % tonne/y

% Calculation of cereals production
flowmat(:,158,6) = cer_prod * crop_a;
% Multiplier
mult = [m_food(2) parause2(153:155)' parause2(156)*1.1622232];
% Composition of cereal
flowmat(:,158,1:5) = flowmat(:,158,6) * mult;   % tonnes/y and kWh/y
flowmat(:,158,7) = flowmat(:,158,1);           % tonnes/y
flowmat(:,158,1) = flowmat(:,158,7)./ 1382000;  % Mgal/d
```

```
% VEGGIES  ************************************************************
% [sweetcorn pumpkin watermelons snapbeans tomatoes]
% Area array = [area values in km2 for as many as veggies considered]
% Total area (295 acres) calculated in 'UC nutrient.xls' for year 2000
% See area distribution for each crop in 'crops inventory 2002.xls'
%veg_area = 295 * 0.0040468564 * [0.60 0.12 0.10 0.09 0.09]; % km2 UCW
veg_area = statout(1,11) * ve_mix(:,2)'/100; %km2 LON

% Vegetable Yield. Original data in CWT/acre UCW
%veg_yield = [138 235 220 48 330]* (0.04535924 / 0.0040468564); % Converted to tonne/km2
veg_yield = ve_mix(:,3)' * parause(200); %tonne/km2.a

% Production after yield losses for first year
veg_prod = sum((veg_area.* veg_yield) * (1 - parause(138))); % tonne/y

% Calculation of vegetables production
flowmat(:,159,6) = veg_prod * crop_a;
% Multiplier
mult = [m_food(3) parause2(157:159)' parause2(160)*1.1622232];
% Composition of vegetable composition
flowmat(:,159,1:5) = flowmat(:,159,6) * mult;   % tonnes/y and kWh/y
flowmat(:,159,7) = flowmat(:,159,1);           % tonnes/y
flowmat(:,159,1) = flowmat(:,159,7)./ 1382000;  % Mgal/d

% OTHERS  ************************************************************
% It is assumed that no significant 'OTHER' production in the Upper
% Chattahoochee Watershed (Alcoholic beverages, Veg. Oil, Sugar,
% Sweeteners, stimulants)
flowmat(:,163,:) = 0;

 %*********************************************************************
% SEEDS AND YIELD LOSSES (as a proportion of production)
%*********************************************************************
for a = 0:2
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
% flowmat(:,254 FRUITS flowmat(:,255 CEREALS flowmat(:,256 VEGGIES
    flowmat(:,254+a,:) = flowmat(:,157+a,:) * parause(138)/(1 - parause(138));
end
% Total yield Losses FRUITS CEREALS VEGGIES
flowmat(:,253,:) = sum(flowmat(:,254:256,:),2);

% Total food delivered (produced) after losses FRUITS CEREALS VEGGIES
flowmat(:,164,:) = sum(flowmat(:,157:163,:),2);

nut_feed

m_crops = cr_mix(:,1)/100;

% Crops for feed in the Upper Chattahoochee portion of Georgia are mainly:
% (in area importance order) corn for silage and hay [cornforsilage hay]
% SPECIFIC FOR UCW (not year specific though)
% Area array = [area values in km2 for as many as crops considered]
% Total area (19500 acres) calculated in 'UC nutrient.xls' for year 2000
%feed_area = 19500 * 0.0040468564 * [0.06 0.94]; % km2 UCW
% feed_area = [year] x [number of feed crops]
feed_area = statout(1,11) * cr_mix(:,2)'/100;
% Yield + uncertainty, in tonne/km2.a
```

```
feed_yield = cr_mix(:,3)'.* parause(201); %tonne/km2.y

% Matrix space allocation
feed_det = zeros(length(m_crops),nmat);

% Production after yield losses for first year in tonne/y
% Calculation of contituents based on composition of Dry Matter d.m.
% This section was coded for only two types of feed crops: corn for silage
% and all hay. If more crops are selected by the user, this section must
% need adjustments.
% -------------------------------------------------------------------------
b = 169; % parause2(169 parause2(170 parause2(171 parause2(172
for a = 1:length(m_crops)
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
    % Total mass for each crop
    feed_det(a,6) = feed_area(a) * feed_yield(a) * (1 - parause(138));
    % Water component of the total mass
    feed_det(a,7) = feed_det(a,6) * m_crops(a);   % tonne/y
    feed_det(a,1) = feed_det(a,7) / 1382000;        % Mgal/d
    % Other constituents
    feed_det(a,2:5) = (feed_det(a,6).* (1 - m_crops(a)).* parause2(b:b+3))';
    b = 173; %parause2(173 parause2(174 parause2(175 parause2(176
end
%--------------------------------------------------------------------------
% Total production and composition for first year
feed_prod = sum(feed_det); % Mgal/d, tonne/y, and kWh/y respectively

% Total production and composition adjusted per year, in tonne/y and kWh/y
% by the change in crop area in time
for a = 1:x % flowmat(:,177,:)
    flowmat(a,177,:) = feed_prod * crop_a(a);
end
% Feed yield losses
flowmat(:,305,:) = flowmat(:,177,:) * parause(138)/(1 - parause(138));

%**************************************************************************
% EXPORTS/IMPORTS OF FOOD/FEED/LIVESTOCK (consumed - produced)
%**************************************************************************
% ** FOOD **
% [fruits cereal veggies]
flowmat(:,165:167,:) = flowmat(:,149:151,:) - flowmat(:,157:159,:);
% [B-meat Po-meat Pi-meat]
E_meat = C_meat - P_meat; % consumed - produced
flowmat(:,168,:) = sum(E_meat,2);
% [eggs milk]
E_dairy = C_dairy - P_dairy;
flowmat(:,170,:) = sum(E_dairy,2);
% [fish others]
flowmat(:,169:2:171,:) = flowmat(:,153:2:155,:) - flowmat(:,161:2:163,:);
% All food
flowmat(:,172,:) = sum(flowmat(:,165:171,:),2);

% ** FEED **
```

```
flowmat(:,178,:) = flowmat(:,176,:) - flowmat(:,177,:);

% ** LIVESTOCK **
% Import/Export factor as a function of:
% growth + selling + breeding
live_ief = live_gf' + parause(158:160) - parause(202:204);

for a = 1:x
% [B Po Pi] flowmat(:,192 flowmat(:,193 flowmat(:,194
% Mass of livestock exported/imported in tonne/a
    flowmat(a,192:194,6) = (livestock(a,:)'.* live_ief).* statin(39:41);
% composition used is same as live animal
    flowmat(a,192:194,7) = flowmat(a,192:194,6).* m_food(4:6);  % tonnes/y
    flowmat(a,192:194,1) = flowmat(a,192:194,7)./ 1382000;      % Mgal/d
    flowmat(a,192:194,2) = flowmat(a,192:194,6).* parause2(189:4:197)'; % tonnes/y
    flowmat(a,192:194,3) = flowmat(a,192:194,6).* parause2(190:4:198)'; % tonnes/y
    flowmat(a,192:194,4) = flowmat(a,192:194,6).* parause2(191:4:199)'; % tonnes/y
    flowmat(a,192:194,5) = flowmat(a,192:194,6).* parause2(192:4:200)' * 1.1622232; % kWh/y

% MODULE Waste Management sector
wastems;

flowmat(:,269,:) = remainTS * parause(194);

% Treated sewage sludge incinerated
flowmat(:,335,:) = remainTS * (1 - parause(194)) * parause(195);

% Treated sewage sludge to composting
flowmat(:,279,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * parause(196);

% Treated sewage sludge exported as fertilizer
flowmat(:,342,:) = -remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * parause(197);

% The remaining goes to landfill
flowmat(:,275,:) = remainTS * (1 - parause(194))  * ...
    (1 - parause(195)) * (1 - parause(196)) * (1 - parause(197));

%*********************************************************************
% WASTE MANAGEMENT: recycling, incineration
%*********************************************************************
mswcalc

flowmat(:,116,:) = flowmat(:,108,:)/(1 - parause(89)); % paper
flowmat(:,117,:) = flowmat(:,109,:)/(1 - parause(90)); % Food
flowmat(:,118,:) = flowmat(:,110,:)/(1 - parause(91)); % Wood
% flowmat(:,119,:)% yard waste generated
% Total yard waste generated is estimated based on how much waste is
% disposed and how much is left on site. Yard waste and grass left on site
% is calculated in 'yardwaste.m' as part of the forestry sector module.
% Variable: 'flowmat(:,119,:)' and 'ywLeft'
%*********************************************************************
% Waste reducing methods
%*********************************************************************
% Recycled/Recovered paper
flowmat(:,112,:) = -1 * (flowmat(:,116,:) - flowmat(:,108,:));

% Recovered food
flowmat(:,113,:) = (flowmat(:,117,:)- flowmat(:,109,:));
% Food sent to anaerobic digestion
```

```
flowmat(:,345,:) = flowmat(:,113,:) * parause(240);
% Remaining food waste sent to composting
flowmat(:,344,:) = flowmat(:,113,:) - flowmat(:,345,:);

% Reused wood
flowmat(:,114,:) = (flowmat(:,118,:)- flowmat(:,110,:));
% Composted yard waste
flowmat(:,115,:) = (flowmat(:,119,:)- flowmat(:,111,:) - ywLeft);

%*********************************************************************
% Anaerobic digestion of food waste
%*********************************************************************
fooddig

FD_par = [parause(270) parause(271) 0.70];

% Volatile solids destroyed in digestion as a function of total solids in
% food waste, tonnes per annum
VS_Food = flowmat(:,345,6) * FD_par(1);
FD_des = VS_Food * FD_par(2); % tonnes per annum

% Ideal gas V = RT/P. This gives a value of 22.414 L at STP ('Standard
% temperature and pressure' of 273.15 K and 1 atm).
% Vtotal = Vch4 + Vco2 => Vch4 * [1/% - 1] = Vco2
% Mch4 = [Vch4 / 22.414] * MWch4
% Volume / mass factor
FD_MtoV = (16 + 44 * (1 - FD_par(3))/FD_par(3))/22.414;
% Volume of methane
FD_metV = FD_des * 1000 / FD_MtoV; % m3 per annum
FD_vol = FD_metV / FD_par(3); % m3 per annum
FD_co2V = FD_vol - FD_metV; % m3 per annum
% typical Biogas yield Litre/g VS 0.6, m3/tonne: 465.4(d.w.), 143.8(w,w.)
% check:
% w.w. FD_vol./flowmat(:,345,6)
% d.w. FD_vol./(flowmat(:,345,6) - flowmat(:,345,7))
% Total carbon in biogas (tonnes per annum)
FD_metM = calmass(1,0,16,FD_metV,0); % [P atm,T C,MW,mass or vol,mod]
FD_co2M = calmass(1,0,44,FD_co2V,0); % [P atm,T C,MW,mass or vol,mod]
FD_car = FD_metM * 12/16 + FD_co2M * 12/44;

%*********************************************************************
% BIOGAS FLOW
%*********************************************************************
% - Biogas collected from Sewage Sludge DIGESTION
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,346,2) = 0;
flowmat(:,346,3) = 0;
flowmat(:,346,4) = FD_car;
flowmat(:,346,5) = FD_metV * 10.343; % kWh/a

%*********************************************************************
% FOOD SLUDGE FLOW
%*********************************************************************
% Calculated as the difference between feed and biogas
flowmat(:,347,1) = flowmat(:,345,7)./ 1382000;
flowmat(:,347,2) = flowmat(:,345,2);
flowmat(:,347,6) = flowmat(:,345,6) - FD_des;
flowmat(:,347,7) = flowmat(:,345,7);
flowmat(:,347,3) = flowmat(:,345,3);
```

```
% Carbon (corrected by VS destruction)
flowmat(:,347,4) = flowmat(:,345,4) - FD_car;

% Energy value (function of ultimate analysis of treated sludge)
% see reference Thipkhunthod 2005, eq 20:
% HHV = 406.4C - 210.6H + 154.7S + 160.3O - 151.3N - 23.8A + 0.0034
% Assumed Ash content 35% (21-43) and Sulphur 1% (0.5-1.5)
% Remaining mass has the form CyHzOtNx
%m_A = 35; % Ash content as a percentage
%m_S = 1; % Sulphur content as a percentage
%num_moles = (100 - m_A - m_S)/ Tx; % number of moles per unit of mass
%m_C = num_moles * Cy * 12;  % Carbon content as a percentage
%m_H = num_moles * Hz * 1;   % as a percentage
%m_O = num_moles * Ot * 16;  % as a percentage
%m_N = num_moles * Nx * 14;  % as a percentage
% HHV in KJ/kg (dry basis)
%flowmat(:,289,5) = 406.4 * m_C - 210.6 * m_H + 154.7 * m_S +160.3 * m_O - 151.3 * m_N - 23.8 * m_A
+ 0.0034;
%HHVkj = flowmat(:,289,5);
% HHV in kWh/year (dry basis)
%flowmat(:,289,5) = flowmat(:,289,5).* SD_Tss * 1000 * 0.0002777780;
flowmat(:,347,5) = flowmat(:,345,5) - flowmat(:,346,5);

% Estimation of materials combusted MSW
% wood combusted(wood/paper/yard)
flowmat(:,125,:) = (flowmat(:,108,:) + flowmat(:,110,:) + flowmat(:,111,:)) * parause(93);
% Food combusted
flowmat(:,258,:) = flowmat(:,109,:) * parause(93);

%**********************************************************************
% Emissions associated to MSW incineration (wood/paper/yard waste)
%**********************************************************************
mswinci

AshC = 0.02;
% [W N P C E M WM]
Combpro = -[1 1-AshC 0 1-AshC 0 1-AshC 1];
for a = 1:nmat
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
```

```
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
% Emission from wood products in MSW incineration
    flowmat(:,143,a) = flowmat(:,125,a) * Combpro(a);
% Emission from food products in MSW incineration
    flowmat(:,280,a) = flowmat(:,258,a) * Combpro(a);
% Emission from sewage sludge in incineration
    flowmat(:,336,a) = flowmat(:,335,a) * Combpro(a);
end
% Energy content of incineration gases, complete
flowmat(:,143,5) = 0;
flowmat(:,280,5) = 0;
flowmat(:,336,5) = 0;

%***********************************************************************
% RECOVERY OF ENERGY FROM MSW COMBUSTION
%***********************************************************************
% Energy generated is calculated based on a the energy content of materials
% and a combustion efficiency
% Energy from MSW (wood + food + sludge)
% Based on efficiency
flowmat(:,257,5) = (flowmat(:,125,5) + flowmat(:,258,5) + flowmat(:,335,5))* parause(217);
% Only sludge
flowmat(:,337,5) = (flowmat(:,335,5))* parause(217);

%***********************************************************************
% CALCULATION OF THE RESIDUAL SOLID LEFT AFTER COMBUSTION
%***********************************************************************
% For this we use the content of volatile components in
MSWfeed     = flowmat(:,258,:) + flowmat(:,125,:) + flowmat(:,335,:);
MSWemission = flowmat(:,143,:) + flowmat(:,280,:) + flowmat(:,336,:);
flowmat(:,276,:) = MSWfeed + MSWemission;

%************************************************************************
% First part of the energy module. second part is 'energys2.m'
%************************************************************************
energys1

energyreq

pop_hou = [statout(:,2)*1000 statout(:,2).*(1000/statin(46))];

% Required energy matrix [year] x [DO CO IN TR] in kWh/year
% Energy after all losses and efficiencies
% Pre-allocates matrix space
req_ener = zeros(x,4);

req_ener(:,1) = pop_hou(:,2) * parause(174) * 1000; % DO
req_ener(:,2) = pop_hou(:,1) * parause(177) * 1000; % CO
```

```
req_ener(:,3) = pop_hou(:,1) * parause(180) * 1000; % IN
req_ener(:,4) = pop_hou(:,1) * parause(184) * 1000; % TR

% Electricity delivered in kWh/year
del_elec = zeros(x,4);      % Pre-allocates matrix space
% [DO CO IN TR]
% Patch. Energy requirement for food griders is added as a function of
% population.
flowmat(:,340,5) = parause(221) * statout(:,2) * 1000 * parause(67);
del_elec(:,1) = req_ener(:,1) * (1 - sum(parause(175:176))) + flowmat(:,340,5);
del_elec(:,2) = req_ener(:,2) * (1 - sum(parause(178:179)));
del_elec(:,3) = req_ener(:,3) * (1 - sum(parause(181:183)));
del_elec(:,4) = req_ener(:,4) * (1 - sum(parause(185:187)));

tol_elec = del_elec;

% Natural gas required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency is 65% in the case of DO, CO,and IN. and 90%
% Efficiency for Natural Gas internal combustion engines.
% With this it is assumed that the data of kWh of energy use reported from
% EIA accounts for these efficiencies
tol_ngas = zeros(x,4);
tol_ngas(:,1) = req_ener(:,1) * parause(175) * (0.65/parause(208));
tol_ngas(:,2) = req_ener(:,2) * parause(178) * (0.65/parause(208));
tol_ngas(:,3) = req_ener(:,3) * parause(181) * (0.65/parause(208));
tol_ngas(:,4) = req_ener(:,4) * parause(185) * (0.90/parause(206));

% Biomass required in kwh/y [year] x [DO CO IN TR]
tol_biom = zeros(x,4);
% Household appliance energy efficiency is accounted for. Base value is
% assumed to be 40% for the wood consumption stated in m3. The same
% efficiency is assumed true for commercial and industrial applications
% Biomass 'flowmat(:,123,5)' calculated in forestrys.m
tol_biom(:,1) = flowmat(:,123,5) * (0.40/parause(207));
tol_biom(:,2) = req_ener(:,2) * parause(179) * (0.40/parause(207));
tol_biom(:,3) = req_ener(:,3) * parause(182) * (0.40/parause(207));
tol_biom(:,4) = req_ener(:,4) * 0;

% Coal required in kwh/y  [year] x [DO CO IN TR]
% Efficiency assumed similar to coal fired power plants: 32%
tol_coal = zeros(x,4);
tol_coal(:,1) = req_ener(:,1) * 0;
tol_coal(:,2) = req_ener(:,2) * 0;
tol_coal(:,3) = req_ener(:,3) * parause(183) * (0.32/parause(167));
tol_coal(:,4) = req_ener(:,4) * 0;

% Diesel required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_dies = zeros(x,4);
tol_dies(:,1) = req_ener(:,1) * parause(176) * (0.25/parause(205));
tol_dies(:,2) = req_ener(:,2) * 0;
tol_dies(:,3) = req_ener(:,3) * 0;
tol_dies(:,4) = req_ener(:,4) * parause(186) * (0.25/parause(205));

% Gasoline required in kwh/y [year] x [DO CO IN TR]
% Assumed base efficiency for data is 0.25
tol_gaso = zeros(x,4);
tol_gaso(:,1) = req_ener(:,1) * 0;
tol_gaso(:,2) = req_ener(:,2) * 0;
tol_gaso(:,3) = req_ener(:,3) * 0;
tol_gaso(:,4) = req_ener(:,4) * parause(187) * (0.25/parause(205));
```

```
%************************************************************************
% TOTALIZING ENERGY FLOWS BY FUEL TYPE (e.g. electricity, natural gas, etc)
%************************************************************************
% Electricity (normal use + SST + WWTP + WTP + Algae + pyrolysis)
flowmat(:,195,5) = sum(tol_elec,2) + flowmat(:,334,5) + flowmat(:,332,5) + ...
    + flowmat(:,333,5) + flowmat(:,339,5) + flowmat(:,338,5); % kWh/y
% Natural Gas (DO, CO, IN, TR)
flowmat(:,196,5) = sum(tol_ngas,2);
% Biomass
flowmat(:,197,5) = sum(tol_biom,2);
% Coal
flowmat(:,198,5) = sum(tol_coal,2);
% Diesel
flowmat(:,199,5) = sum(tol_dies,2);
% Gasoline
flowmat(:,200,5) = sum(tol_gaso,2);
% Alternative 1
%flowmat(:,201,5) = ;
% Alternative 2
%flowmat(:,202,5) = ;
% Alternative 3
%flowmat(:,203,5) = ;
% Total energy Required by the system (all primary fuels) with
% efficiencies accounted for
flowmat(:,297,5) = sum(flowmat(:,196:203,5),2);  % kWh/y

%************************************************************************
% ESTIMATION OF LOCAL BIODIESEL PRODUCTION (inputs and outputs)
%************************************************************************
% Mainly referred to production of biodiesel from biomass
biofuel

flowmat(:,244,:) = flowmat(:,144,:) * parause(68);
% Total input of biomass for biofuels (function of plant Capacity)
% Local + imported (to satisfy biofuel plant capacity)
flowmat(:,245,:) = 0; % FOR NOW zero, meaning there is no biofuel plant

% Imported Biomass for Biofuels
flowmat(:,246,:) = flowmat(:,245,:) - flowmat(:,244,:);

% Check biomass available versus plant capacity
A = sqrt(flowmat(:,246,:));
if isreal(A) == false
    end
end
% Production of biodiesel from BIOMASS (logging residue)
flowmat(:,242,:) = 0;
% Emissions of the Biofuel production process (from biomass)
flowmat(:,243,:) = 0;

%************************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y)
% Local consumption (non power)
%************************************************************************
% Electricity has no physical units so it is not included in this setcion
% Natural Gas, diesel, and gasoline are calculated in energy2
% Biomass (INCLUDES firewood for domestic USE)
% Using wood properties
flowmat(:,197,6) = flowmat(:,197,5)/parause2(132);  % tonne/y
```

```
flowmat(:,197,1) = flowmat(:,197,6) * parause2(133)/ 1382000 ; % Mgal/d
flowmat(:,197,2:4) = flowmat(:,197,6) * parause2(129:131)'; % tonne/y
flowmat(:,197,7) = flowmat(:,197,1)/1382000;

% Coal (10% Moisture)
flowmat(:,198,6) = flowmat(:,198,5)/parause2(241);  % dry basis tonne/y
flowmat(:,198,1) = flowmat(:,198,6) * (0.1/0.9) / 1382000 ; % Mgal/d
flowmat(:,198,2:4) = flowmat(:,198,6) * (parause2(238:240)/100)'; % tonne/y
flowmat(:,198,7) = flowmat(:,198,1)/1382000;
flowmat(:,198,6) = flowmat(:,198,6) + flowmat(:,198,7); % wet basis tonne/y

% Alternative f1
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f2
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
% Alternative f3
%flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
%flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
%*********************************************************************
% Estimation of flows for power generation
%*********************************************************************
% 1 year = 8760 hours. We calculate kWh/y based on generation capacity
% (MW) by multiplying the number of hours in a year.
% Columns [Coal NG Nuclear      Diesel  Hydro Alt1      Alt2 Alt3]
elec_gen = statout(:,14:21) * 8760 * 1000; % Converted from MW to kWh/y
% Fuel based (nuclear and hydro are assumed that need no fuel flow)
flowmat(:,260,5) = sum(elec_gen,2) - elec_gen(:,3) - elec_gen(:,5); % in kWh/y
% Hydro power
flowmat(:,261,5) = elec_gen(:,5); % in kWh/y

% Power generation time series for Geo and Wind
for a = 0:1 % geo or wind
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
    for b = 1:x % year
        if b == 1
            statout(b,38+a) = inout(b,17+a) + statin(50+a);
        end
    end
end
% Geo power
flowmat(:,262,5) = statout(:,38) * 8760 * 1000; % in kWh/y
% Wind power
flowmat(:,263,5) = statout(:,39) * 8760 * 1000; % in kWh/y

%*********************************************************************
% ESTIMATION OF FUEL CONSUMPTION FOR POWER GENERATION (only coal)
%*********************************************************************
% Natural gas and diesel is completed in energy2.m
% Matrix containing all efficiencies for power generation for each fuel
% type: [Coal NG Nuclear Diesel Hydro Altp1     Altp2 Altp3]. However, we
% eliminate 'Hydro' and 'Nuclear' from this calculation as they are assumed
% to need no fuel.
% Efficiency power generation [Coal NG Diesel Alt-p1 Alt-p2 Alt-p3]
effi_gen = parause(167:172);    % Fraction of the fuel energy value that is
                               % successfully transformed to electricity
elec_gen(:,3) = [];             % Nuclear is removed from matrix
elec_gen(:,4) = [];             % Hydro is removed from matrix
```

```
% flowmat(:,206 flowmat(:,207 flowmat(:,208 flowmat(:,209 flowmat(:,210
% flowmat(:,211
for a = 1:x
    % For all fuel sources
    orig_gen = elec_gen(a,:)./effi_gen';
    % In case some efficiencies are introduced as zero '0' we replace NaN
    % by zeros (prevents errors further down)
    nanG = find(isnan(orig_gen));
    orig_gen(nanG) = zeros(size(nanG));
    flowmat(a,206:211,5) = orig_gen;

    % Only for Alt p1, Alt p2, and Atl p3
    flowmat(a,209:211,6) = flowmat(a,209:211,5)./ parause2(253:4:261)'; % tonne/y
    flowmat(a,209:211,2) = flowmat(a,209:211,6).* parause2(250:4:258)'/100; % tonne/y
    flowmat(a,209:211,3) = flowmat(a,209:211,6).* parause2(251:4:259)'/100; % tonne/y
    flowmat(a,209:211,4) = flowmat(a,209:211,6).* parause2(252:4:260)'/100; % tonne/y
end
% Coal [assumed moisture 10%]
flowmat(:,206,6) = flowmat(:,206,5)./ parause2(241);        % tonne/y dm
flowmat(:,206,7) = flowmat(:,206,6).* 0.10;                 % tonne/y
flowmat(:,206,1) = flowmat(:,206,7)./ 1382000;              % Mgal/d
flowmat(:,206,2:4) = flowmat(:,206,6) * (parause2(238:240)/100)'; % tonne/y

%**********************************************************************
% CALCULATIONS OF Coal Combustion Products (CCP)
%**********************************************************************
ccpcalc

ccratio = 0.325;

% The model contemplates two coal uses: (i) POWER GENERATION, AND (ii)
% OTHERS (DO + CO + IN + TR). CCP will be calculated for each use and added
% up to report the total.
% power and others uses: DO CO IN TR
ccpPO = zeros(x,2,nmat); % [years x coal uses (Power & Other) x species]
ccpPO(:,:,6) = [flowmat(:,206,6) flowmat(:,198,6)] * ccratio;
% Water in CCP. Moisture generally 1 - 3%
ccpPO(:,:,7) = ccpPO(:,:,6) * 0.02;
ccpPO(:,:,1) = ccpPO(:,:,7)./ 1382000;  % Mgal/d
% Energy in CCP. Generally an energy content HHV = 6 MJ/kg
ccpPO(:,:,5) = ccpPO(:,:,6) * 6 * 1000000 * 0.000277778; % kWh/y

% nitrogen left in CCP is calculated USING Baxter 1996, "Nitrogen release
% during coal combustion"  that presents N loss versus Mass loss of coal.
% Also see notebook page 77. The mass loss in N is proportional to
% the overall mass loss (in dry basis) of carbon -> CCP but smaller by
% a ratio 1.2 - 1.5. We use a ratio of 1.35
Nfactor = ccratio/1.35;     % Final nitrogen content in CCP over nitrogen
                           % content in raw coal (dry basis)
                           % Nitrogen in CCP
ccpPO(:,:,2) = [flowmat(:,206,2) flowmat(:,198,2)] * Nfactor;

% Phosphorus in CCP. Mahowald 2008 reports that about 0.5-1% of particles
% released from combustion is P for power applications. Coal combustion
% releases about 1%, waste incineration is about 0.6%.
% Emission of particles is taken from Mikael 2000 (1 - 30 mg per MJ). Using
% 15 mg per MJ or (15 * 0.27777 / 10^9 tonne per kwh)
emittedP = [flowmat(:,206,5) flowmat(:,198,5)] * 15 * 0.27777 / 10^9;
emittedP = emittedP.* (ones(x,1) * [0.01 0.006]);
```

```
ccpPO(:,:,3) = [flowmat(:,206,3) flowmat(:,198,3)] - emittedP;

% Calculation of Carbon in CCP [power others]: Potential CO2 emissions are
% typically in the order of 205 lbs CO2 per MMbtu (Hong 1994). Roy 2009
% estimated different equations for CO2 emmissions, resulting in about
% 1.45 kg CO2 per kg of coal. IPPC resports a value of 25.8 t C per TJ. In
% all these cases CO2 emissions correspond to close to full oxidation of
% Carbon in Coal. Hower 2005 reports less than 3% of Carbon (ultime
% analysis) in bottom and slag ash. We will use 2% as the CCP carbon
% content.
ccpPO(:,:,4) = ccpPO(:,:,6) * 0.02;

flowmat(:,319,:) = ccpPO(:,1,:);

% Totals for CCP for both POWER and OTHER applications. See previous
% calculations in 'ccpcalc.m'
flowmat(:,287,:) = sum(ccpPO,2);

% CPP sent outside the system
flowmat(:,291,:) = -1 * flowmat(:,287,:) * (1 - parause(83));

%************************************************************************
% Emissions from power generation (coal)
%************************************************************************
% [coal NG diesel Altp1 Altp2 Altp3]
% Emissions per fuel type in tonne/y
% Natural gas and diesel are calculated in energy2.m
% Coal
flowmat(:,212,:) = -1 * (flowmat(:,206,:) - ccpPO(:,1,:));
flowmat(:,212,5) = 0; % No energy content in emission

%************************************************************************
% Algae growing from coal plant emissions
%************************************************************************
% Recovery of nutrient from Coal combustion flue gas (mod 12/2010)
% Matrix of necessary nutrients to achieve the desired algae growth
FertAlg = zeros(x,1,nmat);
% Algae process
if parause(215) > 0 || parause(65) > 0
    AlgaeC

NutInt2 = modset(8); % If 1 then N, if 2 then P, if 3 then C

    % Needed fertilizer from external sources is calculated (N, P and C)
    FertAlg(:,1,2:4) = -1 * flowmat(:,212,NutInt2+1) * parause(215) * aAl./aAl(NutInt2);
    FertAlg(:,1,2:4) = FertAlg(:,1,2:4) + flowmat(:,212,2:4) * parause(215);

    % Flow of nutrients captured from flue gas
    ExFertAlg = FertAlg;
    ExFertAlg(ExFertAlg(:,1,:) > 0) = 0;
    flowmat(:,326,2:4) = -1 * flowmat(:,212,2:4) * parause(215) + ExFertAlg(:,1,2:4);

    % We eliminate negative values from FertAlg matrix (which means that there
    % is excess of this particular nutrient)
    FertAlg(FertAlg(:,1,:) < 0) = 0;

% Total mass of algae produced (based on nutrient of interest) aAl(NutInt)/TAx
% From AlgaeW too
T_algae = -1 * flowmat(:,212,NutInt2+1) * parause(215) * TAx/aAl(NutInt2) + ...
    T_algae;
```

```
%************************************************************************
% Production of biodiesel
%************************************************************************
% Lipid cells (in algae on a dry basis) are assumed an atomic formula
% similar to CLy HLz OLt NLx (no P, S, Ash)
AUlt(2,:) = [12*parause2(39) 1*parause2(38) 16*parause2(40) ...
    14*parause2(37) 31*0 32*0 0];
%
% Total atomic mass of lipid portion in algae cells
TLx = sum(AUlt(2,:));

% Lipids content of algae biomass (percentage)
Lipids = parause(216);
%Biodiesel from ALGAE (dry)
flowmat(:,304,6) = T_algae.* Lipids;
% water is assumed to be removed and esterification has being performed
flowmat(:,304,7) = 0;
% Nitrogen
flowmat(:,304,2) = flowmat(:,304,6) * AUlt(2,4)/TLx;
% Phosphorus in biodiesel (usually 0)
flowmat(:,304,3) = flowmat(:,304,6) * AUlt(2,5)/TLx;
% Carbon
flowmat(:,304,4) = flowmat(:,304,6) * AUlt(2,1)/TLx;

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Assumed ash content of algae oil (percentage of dry mass)
aL_A = 0;
% Hydrogen content in oil (percentage)
aL_H = AUlt(2,2)/TLx * 100;
% Oxygen content in oil (percentage)
aL_O = AUlt(2,3)/TLx * 100;
% Carbon content in oil (percentage)
aL_C = AUlt(2,1)/TLx * 100;
% Nitrogen content in oil (percentage)
aL_N = AUlt(2,4)/TLx * 100;
% No sulphur
aL_S = AUlt(2,6)/TLx * 100;

% HHV in MJ/kg of dry mass
alLener = 0.3491 * aL_C + 1.1783 * aL_H + 0.1005 * aL_S * ones(size(x,1),1)...
    - 0.1034 * aL_O * ones(size(x,1),1) - 0.015 * aL_N ...
    - 0.0211 * aL_A * ones(size(x,1),1);
% Energy in kWh/y (dry basis)
flowmat(:,304,5) = alLener.* flowmat(:,304,6) * 1000 * 0.277778;

% Energy usage by the algae production process
% as a function of the mass of bio-oil produced
flowmat(:,339,5) = parause(220) * flowmat(:,304,6) * 1000;

%************************************************************************
% Production of algae biomass for other uses
% FOR NOW, algae biomass is sent to pyrolysis. update. check. complete
%************************************************************************
% Remaining biomass (after biodiesel is extracted) <-- sent to pyrolysis
flowmat(:,325,6) = T_algae.* (1 - Lipids);
% Corrected mass by Ash content
T_biomass = flowmat(:,325,6);
flowmat(:,325,6) = flowmat(:,325,6) * 100/(100 - a_A);
```

```
% Algae biomass (after oil extraction on a dry basis) are assumed an atomic
% formula similar to CByHBzOBtNBxPBtS.
% First we estimate the number of moles of Algae and Lipids
molA = T_algae/TAx;
molL = flowmat(:,304,6)/TLx;
% Now we calculate the moles of biomass, but since it is already is
% multiplied by the molecular weight of each element, then AUlt for biomass
% contains kilograms.
AUlt(3:3+x-1,:) = molA * AUlt(1,:) - molL * AUlt(2,:);
%
% assumed that water has being removed and esterification has being performed
flowmat(:,325,7) = 0;
% Nitrogen
flowmat(:,325,2) = AUlt(3:3+x-1,4);
% Phosphorus
flowmat(:,325,3) = AUlt(3:3+x-1,5);
% Carbon
flowmat(:,325,4) = AUlt(3:3+x-1,1);

% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Hydrogen content in BIOMASS (percentage) after oil extraction
a_H = AUlt(3,2)/sum(AUlt(3,:)) * 100;
% Oxygen content in BIOMASS (percentage) after oil extraction
a_O = AUlt(3,3)/sum(AUlt(3,:)) * 100;
% Carbon content in BIOMASS (percentage) after oil extraction
a_C = AUlt(3,1)/sum(AUlt(3,:)) * 100;
% Nitrogen content in BIOMASS (percentage) after oil extraction
a_N = AUlt(3,4)/sum(AUlt(3,:)) * 100;
% Sulphur
a_S = AUlt(3,6)/sum(AUlt(3,:)) * 100;

% HHV in MJ/kg of dry mass (substances content as percentage)
alBener = 0.3491 * a_C + 1.1783 * a_H + 0.1005 * a_S * ones(size(x,1),1)...
    - 0.1034 * a_O * ones(size(x,1),1) - 0.015 * a_N ...
    - 0.0211 * a_A * ones(size(x,1),1);
% Energy in kWh/y
flowmat(:,325,5) = alBener.* flowmat(:,325,6) * 1000 * 0.277778;
end
if parause(215) <= 0 && parause(65) <= 0
end
% Emissions from COAL COMBUSTION corrected with the amount captured for
% algae growing 'AlgaeC'
flowmat(:,212,:) = flowmat(:,212,:) + flowmat(:,326,:);

%*********************************************************************
% ESTIMATION OF energy IMPORTS/EXPORTS (consumed - produced)
%*********************************************************************
% Negative values reflect an export flow (outbound)
% Natural gas and diesel are calculated in energy2.m
% 205 Imports/Exports of Electricity
% Calculated in energy2.m
% 233   Imports/exports of coal (None produced locally)
flowmat(:,233,:) = (flowmat(:,198,:) + flowmat(:,206,:)) - 0;

% 236   Imports/Exports of Biomass (for DO, CO, IN, TR energy purposes)
% If wood produced locally (rounwood) is in excess compared to local
% consumption plus output of roundwood then this excess is considered a
% local source of wood for energy purposes
```

```
Excess = 0;
if flowmat(:,106,6) < 0
end
% local logging residue for energy (combustion)
flowmat(:,281,:) = flowmat(:,144,:) * (1 - parause(68));
% Local sources: logging residue, sawmill residue, Excess forestry industry
local_biomass = flowmat(:,281,:) + flowmat(:,120,:) + Excess;

% Imports/Exports of Biomass (energy purposes, firewood and combustion)
% Based on total mass
flowmat(:,236,:) = flowmat(:,197,:) - local_biomass;

%**********************************************************************
% CALCULATIONS OF THE PYROLYSIS PROCESS
%**********************************************************************
% Animal manure sent to waste management, i.e. not used for fertilization,
% has two options: pyrolysis process or composting
% Manure sent to pyrolysis
flowmat(:,348,:) = flowmat(:,188,:) * parause(241); % cattle
flowmat(:,270,:) = flowmat(:,189,:) * parause(193); % poultry
flowmat(:,350,:) = flowmat(:,190,:) * parause(243); % swine
flowmat(:,353,:) = flowmat(:,348,:) + flowmat(:,270,:) + flowmat(:,350,:);

%**********************************************************************
% SOME properties of poultry litter (for pyrolysis process)
% Before searching for more information, it is assumed that cattle and
% swine manure have similar properties
%**********************************************************************
% Assumed ash content of poultry (percentage)
m_A_pl  = 20;
% Calculation of H and O in poultry litter (only if sent to pyrolysis)
if parause(193) == 0
       m_H_pl = 0;
       m_O_pl = 0;
end
% Executes 'Pyro' if sludge - poultry - Algae is sent to pyrolysis
% Flag
if parause(194) > 0 || parause(193) > 0 || parause(215) > 0 || parause(65) > 0
    pyro

py_feed = flowmat(:,270,:) + flowmat(:,269,:) + flowmat(:,325,:) + ...
    flowmat(:,348,:) + flowmat(:,350,:);
flowmat(:,268,:) = py_feed;
% Estimation of the ratio of each feed (manure+sewage+algae biomass) on a
% dry basis. 12/2010
dryMass = [ flowmat(:,270,6)-flowmat(:,270,7) ...
    flowmat(:,269,6)-flowmat(:,269,7) flowmat(:,325,6)-flowmat(:,325,7)...
    flowmat(:,348,6)-flowmat(:,348,7) flowmat(:,350,6)-flowmat(:,350,7)];

% Dry mass of total feed
DryFE = (py_feed(:,:,6) - py_feed(:,:,7)) * ones(size(dryMass,2),1)';
% Mass ratio of each feed type
py_ratio = dryMass./DryFE; % The row sum must be ONE. check
% Ash, Hydrogen, and Oxygen in feed
% Assumed values for cattle and swine manure from
% Proximate and ultimate analysis results for animal manures.pdf
m_A_ca = 13.4; m_H_ca = 4.6; m_O_ca = 41.6;
m_A_sw = 17.7; m_H_sw = 5.6; m_O_sw = 38.8;

AHO =  [m_A_pl m_A a_A m_A_ca m_A_sw
```

```
        m_H_pl m_H a_H m_H_ca m_H_sw
        m_O_pl m_O a_O m_O_ca m_O_sw];

% Ash is calculated and added (fraction of Ash dry basis)
Ashf = sum(py_ratio(1,:).* AHO(1,:)/100);
% Hydrogen calculated from feed composition
Hydf = sum(py_ratio(1,:).* AHO(2,:)/100);
% Oxygen calculated from feed composition
Oxyf = sum(py_ratio(1,:).* AHO(3,:)/100);

%************************************************************************
% Preliminary calculations of Oxygen and Hydrogen content
%************************************************************************
% Moisture of pyrolysis feed (original feed with no ash)
OrigM = py_feed(:,:,7)./ py_feed(:,:,6);
% Desirable moisture is assumed to be 10%. Excess water is evaporated
% in an assumed drying process before pyrolysis
py_feed(:,:,7) = DryFE(:,1) * 0.10/( 1 - 0.10);
py_feed(:,:,1) = py_feed(:,:,7)./ 1382000;
% Total feed flow is corrected with the 10% moisture
py_feed(:,:,6) = DryFE(:,1) + py_feed(:,:,7);

% C(dm) content in feed (free of ash and dry)
%py_feed(:,:,4)./ (py_feed(:,:,6) - py_feed(:,:,7))
% N(dm) content in feed (free of ash and dry)
%py_feed(:,:,2)./ (py_feed(:,:,6) - py_feed(:,:,7))
% C(dm) content in sewage treated sludge
%flowmat(:,269,4)./(flowmat(:,269,6) - flowmat(:,269,7))
%%%%%%% ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%%%%%%%
% Calculation of yield factors for each substance
% W, N, P, and C yield factors where estimated based on Singh 2008, see
% notebook pp 101. Total mass is calculated as the sum of: W N C H S O Ash
% [condensate solids] x [W N P C H S O Ash E dmass]
pyYield = zeros(2,10);
% [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
pyFlows = zeros(x,10,4);

% Total mass partition and species partition is simulated based on first
% order kinetics
% Mass partition based on first order kinetics, assuming T = 773.15 K
% (500 C) and using same pre-exponential constant(A) as presented by
% Di Blasi 2001 but adjusting the activation energy parameter (E).
% The Arrehnius equation is used k = A.exp(-E/RT) to estimate kinetic rate
% constants. See calculations in 'UC nutrients.xls'
Treact = (statin(54) + 273.15); % [K]
Rgas = 0.0083145; % [KJ/mol*K]
% Kinetic constant is kept the same for all {liq char gas]
Aki = [exp(23.1) exp(15.0) exp(22.2) ]' * ones(1,10); %[s^-1]
Eact = [         % [Liquid Char Gas] x [W N P C H S O Ash E dmass]
160     158     0       163     161     155     161     0       0       168
104     114     0       111     123     110     129     0       0       113
168     164     0       161     152     170     145     0       0       160
];
% Reaction rates [Liquid Char Gas]
kreact = Aki.* exp(-Eact./(Rgas.* Treact));
Overallk = ones(3,1) * sum(kreact,1);
% mass partition = feed * ratio of reaction rate
pyYield = kreact./Overallk ; %check all 1, sum(pyYield,1)
% Correction for P, Ash, E
% The values for P are accomodated so that 100% of phosphorus is in Char
% The values for Ash also assume that 100% is partitioned to the Char phase
```

```
% pyYield = [condensate solids gas] x [W N P C H S O Ash E dmass]
pyYield(:,[3 8]) = [[0 1 0]' [0 1 0]'];
pyYield(:,9) = [0 0 0]';

% Feed flows [W N P C H S O Ash E dmass]
% for W N P C
pyFlows(:,1:4,1) = [py_feed(:,1,7) py_feed(:,1,2) py_feed(:,1,3) py_feed(:,1,4)];
% Dry mass
pyFlows(:,10,1) = py_feed(:,:,6) - py_feed(:,:,7);
% Ash from py_ratio and respective values in FEED
pyFlows(:,8,1) = pyFlows(:,10,1).* Ashf;
% Hydrogen from py_ratio and respective values in FEED
pyFlows(:,5,1) = pyFlows(:,10,1).* Hydf;
% S assumed same as sludge (ranges always less than 1% of dry matter)
pyFlows(:,6,1) = pyFlows(:,10,1) * m_S/100;
% Oxygen from py_ratio and respective values in FEED
pyFlows(:,7,1) = pyFlows(:,10,1).* Oxyf;

% Checking: sum(pyFlows(:,2:8,1),2) should be near pyFlows(:,10,1)
%%%%%%% END of ESTIMATION OF PARTITION and YIELD FACTORS %%%%%%%%%%%%%%%%%%
for a = 1:x
% Biofuel (biodiesel) production
    pyFlows(a,:,2) = pyFlows(a,:,1).* pyYield(1,:);
% Solids (Char) generation
    pyFlows(a,:,3) = pyFlows(a,:,1).* pyYield(2,:);
end
% Biogas (gas) generation - by difference
pyFlows(:,:,4) = pyFlows(:,:,1) - pyFlows(:,:,2) - pyFlows(:,:,3);

% Energy content for each pyrolysis product (MJ/kg) - Preliminary value
pyenergy = zeros(x,4); % [years] x [feed condensate solids gas]

% Energy content calculated based on Channiwala 2002 (HHV from ultimate
% analysis of gaseous, liquid, and solid fuels)
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% pyFlows = [years] x [W N P C H S O Ash E dmass] x [feed condensate solids gas]
% [feed condensate solids gas]
for a = 1:4
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
```

```
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
% Dry mass based on ultimate composition
    pyDry = sum(pyFlows(:,2:8,a),2) * ones(1,6); % tonnes/y
% Ultimate composition on a dry basis (percentage)
    pyDryC = 100* [pyFlows(:,4,a) pyFlows(:,5,a) pyFlows(:,6,a) ...
        pyFlows(:,7,a) pyFlows(:,2,a) pyFlows(:,8,a)]./pyDry;
% HHV in MJ/kg of dry mass
    pyenergy(:,a) = 0.3491 * pyDryC(:,1) + 1.1783 * pyDryC(:,2) ...
        + 0.1005 * pyDryC(:,3) - 0.1034 * pyDryC(:,4)...
        - 0.015*pyDryC(:,5) - 0.0211*pyDryC(:,6);
% Energy in kWh/y
    pyFlows(:,9,a) = pyenergy(:,a).* pyDry(:,1) * 1000 * 0.277778;
end
%Check energy by mass partition
% [condensate solids gas]
%El = pyenergy(:,2) * pyYield(1,10);
%Ec = pyenergy(:,3) * pyYield(2,10);
%Eg = pyenergy(:,4) * pyYield(3,10);
%El + Ec + Eg;
%alternative for calculating mass of each phase (using mass yield factors)
% [condensate solids gas]
pyFlows(:,10,2) = pyFlows(:,10,1) * pyYield(1,10);
pyFlows(:,10,3) = pyFlows(:,10,1) * pyYield(2,10);
pyFlows(:,10,4) = pyFlows(:,10,1) * pyYield(3,10);

% Checking: sum(pyFlows(:,1:8,1),2)
% sum(pyFlows(:,2:8,2),2)+sum(pyFlows(:,2:8,3),2)+sum(pyFlows(:,2:8,4),2)
% pyFlows(:,10,1) + py_feed(:,:,7)
% Pyrolysis outputs (in flowmat format)
% Water content in mass units tonne/y (assumed to be lost during the
% pyrolysis and upgrading process)
flowmat(:,265,7)    = 0;                    % Liquid phase
flowmat(:,267,7)    = 0;                    % Char phase
flowmat(:,266,7)    = 0;                    % gas phase
% Water content in volume units Mgal/d
flowmat(:,265,1)    = flowmat(:,265,7)./ 1382000;         % Liquid phase
flowmat(:,267,1)    = flowmat(:,267,7)./ 1382000;         % Char phase
flowmat(:,266,1)    = flowmat(:,266,7)./ 1382000;         % gas phase
%N, P, and C in mass units tonne/y
for a = 2:4
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
    flowmat(:,265,a) = pyFlows(:,a,2);                % Liquid phase
    flowmat(:,267,a) = pyFlows(:,a,3);                % Char phase
    flowmat(:,266,a) = pyFlows(:,a,4);                % gas phase
end
% Total mass tonne/y, Liquid, char, gas (NO water)
```

```
flowmat(:,265,6)    = pyFlows(:,10,2);
flowmat(:,267,6)    = pyFlows(:,10,3);
flowmat(:,266,6)    = pyFlows(:,10,4);
% Energy flow kwh/y
flowmat(:,265,5)    = pyFlows(:,9,2);                    % Liquid phase
flowmat(:,267,5)    = pyFlows(:,9,3);                    % Char phase
flowmat(:,266,5)    = pyFlows(:,9,4);                    % gas phase

% check total mass
%sum(pyFlows(:,10,2:4),3) pyFlows(:,10,1)
% Emissions from the pyrolysis and upgrading process
% All the water in feed
%flowmat(:,303,7) = flowmat(:,268,7);
flowmat(:,303,7) = DryFE(:,1)./(ones(x,1)./OrigM - ones(x,1));
flowmat(:,303,1) = flowmat(:,303,7)./ 1382000;

% Energy usage by the pyrolysis process (electricity terms) based on Reed
% XXXX, assuming 1.5 kJ/gram of biomass feed (0.000417 kWh/g)
% as a function of the feed dry mass
%flowmat(:,338,5) = pyFlows(:,10,1) * 1000000 * 0.000417;
% as a function of the mass of bio-oil produced (kWh per kg)
flowmat(:,338,5) = parause(219) * flowmat(:,265,6) * 1000;
end
%***********************************************************************
% TOTAL LOCALLY PRODUCED FERTILIZER (Inorganic)
% Struvite process + Struvite process + Pyrolysis solids
%***********************************************************************
flowmat(:,293,:) = struvite + AmmSulph + flowmat(:,267,:);

%***********************************************************************
% Total production of biodiesel
% (includes BIOMASS (logging residue), PYROLYSIS, and ALGAE)
%***********************************************************************
flowmat(:,264,:) = flowmat(:,242,:) + flowmat(:,265,:) + flowmat(:,304,:);

%***********************************************************************
% LANDFILLING
%***********************************************************************
% Estimation of materials landfilled MSW
% Wood products ladfilled:
% + paper disposed (1 - incinerated)
% + wood disposed (1 - incinerated)
% + yard waste disposed (1 - incinerated)
% + waste from paper production (papermill)
flowmat(:,126,:) = (flowmat(:,108,:) + flowmat(:,110,:) + ...
    flowmat(:,111,:)) * (1 - parause(93)) + flowmat(:,329,:);
% Food lanfilled
flowmat(:,259,:) = flowmat(:,109,:) * (1 - parause(93));
% CPP landfilled
flowmat(:,290,:) = flowmat(:,287,:) * parause(83);
% Total material landfilled
% sludge + food + wood + MSW incineration residual + CCP
flowmat(:,271,:) = flowmat(:,275,:) + flowmat(:,259,:) + ...
    flowmat(:,126,:) + flowmat(:,276,:) + flowmat(:,290,:);

% Content of nutrients in refuse, tonne/y
Refuse = zeros(x,x,3); % [years] x [years] x [N P C]
% We build triangular matrixes for each species
for a = 2:4 % [N P C]
    for b = 1:x % [years]
```

```
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    for b = 1:x % [years]
        tempV = flowmat(b,271,a) * ones(1,x + 1 - b);
        tempA = diag(tempV,b - 1);
        Refuse(:,:,a - 1) = Refuse(:,:,a - 1) + tempA';
    end
end
% Calculates the landfill leaching factor (as a mass fraction) for N, P, C
% Equation for N and C are based on Raveh 1979 (see notes p.91-93). P based
% on estimations reported in notebook p.93-96.
Kleach = -1 * [0.29*0.54.^((1:x)') 0.00058*0.54.^((1:x)')  0.09*0.51.^((1:x)')];

% Kleach is corrected by the ratio of actual precipitation and the long
% term precipitation and then calculate leaching flows from landfill
for a = 2:4  % [N P C]
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
    Kleach(:,a - 1) = Kleach(:,a - 1).* (inout(:,1)/statin(27));
    flowmat(:,299,a) = Refuse(:,:,a - 1) * Kleach(:,a - 1);
end
% Calculates the landfill emission factor (as a mass fraction) for N, P, C
% P emission is zero. Landfill gas is assumed to have a carbon ratio:
metP    = 0.55; % fraction v/v of methane
co2P    = 0.43; % fraction v/v of carbon dioxide
% Nitrogen is not more than the following volume fraction
nitP    = 0.02; % fraction v/v of nitrogen gas
% (see p.94 for logic)
% The Equation used for CH4 calculation first is the one used by LandGEM
% (application by EPA), and from that point the rest of the gases are
% calculated. We use "total amount of MSW disposed" as per EPA
% recommendations for LandGEM application
% time step (within the year)
tst     = (0:0.1:0.9)';
% Assumed parameters (LandGEM)
kmeth   = 0.05;     % Methane generation rate, year^-1
L0      = 170;      % Potential CH4 generation (m3/Mg), Mg = tonne

LF_metV = zeros(2*x-1,x);
stY = 1;
for a = 0:x-1
    for b = 0:x-1;
        LF_metV(b+stY,a+1) = sum(kmeth * L0 *...
            flowmat(a+1,107,6)/10 * exp((b+tst).*-kmeth));
    end
    stY = stY + 1;
end
if x > 1
end
% Calculation of volume of the rest constituents: N2 and CO2, m3/y
LF_TotV = LF_metV / metP;
LF_co2V = LF_TotV * co2P;
LF_nitV = LF_TotV * nitP;
```

```
% Mass of all constituents released (tonne/y)
LF_met = calmass(1,0,16,LF_metV,0); % [P,T,MW,mass or vol,mod]
LF_co2 = calmass(1,0,44,LF_co2V,0); % [P,T,MW,mass or vol,mod]
LF_nit = calmass(1,0,28,LF_nitV,0); % [P,T,MW,mass or vol,mod]

% Total carbon C mass, emissions generated, tonne/y
flowmat(:,272,4) = LF_met * 12/16 + LF_co2 * 12/44;
% Total nitrogen N mass, emissions generated, tonne/y
flowmat(:,272,2) = LF_nit * 14/28; %Good for one year simulation
%flowmat(:,272,2) = flowmat(:,271,2) * 0.45;
% Methane is assumed to have a heating value of 1000 BTU/ft3 at standard
% conditions (0 C and 1 atm) or 10.343 kWh/m3.
flowmat(:,272,5) = LF_metV * 10.343; % kWh/year

% CORRECTION to account for the capture of biogas
% Biogas captured and sent for energy uses
flowmat(:,273,:) =  flowmat(:,272,:).* parause(209);
% Air Emission released from landfill site
flowmat(:,272,:) = -1 * (flowmat(:,272,:) - flowmat(:,273,:));

% Actual material that stays in landfill after emissions and leaching
flowmat(:,301,:) = flowmat(:,271,:) - flowmat(:,273,:) + flowmat(:,272,:)...
    + flowmat(:,299,:);
% Material accumulated in landfill after emissions and leaching
% flowmat(:,302,:)
for a = 1:x
    flowmat(a,302,:) = sum(flowmat(1:a,301,:), 1);
end
% Check that accumulation is always positive
% flowmat(2:x,302,4) - flowmat(1:x-1,302,4)
if isreal(sqrt(flowmat(2:x,302,:) - flowmat(1:x-1,302,:)))  == false
    end
else
    emat(17) = 0;
end
%*********************************************************************
% ESTIMATION OF LOCAL BIOGAS PRODUCTION
%*********************************************************************
% Gas generated from:
% pyrolysis, landfills, sludge digestion, food digestion
% are added together and sent for power generation
flowmat(:,278,:) = flowmat(:,266,:) + flowmat(:,273,:) + ...
    flowmat(:,277,:) + flowmat(:,346,:);

%*********************************************************************
% PRODUCTION OF FERTILIZER  - THE COMPOSTING PROCESS
%*********************************************************************
% INFO SOURCE: Eghball 1997, Tiquia 2002
% Distribution of manure to different waste management practices:
% (a) pryrolysis, (b) composting, (c) Exported
% Manure sento to composting (corrected by the amount sent to pyrolysis)
flowmat(:,349,:) = flowmat(:,188,:) * (1 - parause(241)) * parause(242); % cattle
flowmat(:,351,:) = flowmat(:,189,:) * (1 - parause(193)) * parause(192); % poultry
flowmat(:,352,:) = flowmat(:,190,:) * (1 - parause(243)) * parause(244); % swine
% Animal Manure sent to composting: cattle + poultry + swine
flowmat(:,288,:) = flowmat(:,349,:) + flowmat(:,351,:) + flowmat(:,352,:);
% Animal manure exported (not sent to pyrolysis, not sent to composting)
ManureUsed = flowmat(:,288,:) + flowmat(:,353,:);
flowmat(:,312,:) = -(flowmat(:,191,:) - ManureUsed);
```

```
% COMPOSTED MATERIAL IS ASSUMED TO HAVE DIFFERENT RECOVERY FACTORS FOR EACH
% NUTRIENT
%
% recovery factors of composting, the rest is emitted as an atmospheric
% emission or leached to the soil, for each nutrient [W N P C E Mass]
% (a) Emission factors / negative value (outgoing flow). Energy flow
% assumed to be proportional to mass loss
comEmi = -1 * ones(1,x)' * [0.52 0.35 0 0.50 0 0 0.52];
% (b) Leaching factors (corrected by long term rainfall)
comLea = -1 * (inout(:,1)/statin(27)) * [0 0.018 0.02 0.01 0 0 0];
% (c) Composting factors (fertilizer value)
comFac = ones(x,nmat) + (comLea + comEmi);

% The sum of: yard waste + MSW food + Manure + Treated Sewage Sludge
comp_in = flowmat(:,115,:) + flowmat(:,344,:) + flowmat(:,288,:) + flowmat(:,279,:);
%flowmat(:,344,4)/flowmat(:,344,2)
%flowmat(:,279,4)/flowmat(:,279,2)
% Check that the C/N ratio is between 10 and 30
CtoN = comp_in(:,1,4)./ comp_in(:,1,2);
if any(CtoN < 10 | CtoN > 30) == true
    end
end
for a = 1:nmat
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
    % Emission to the atmosphere
    flowmat(:,298,a) = comp_in(:,1,a).* comEmi(:,a);
```

```
    % Leaching to hydrosphere
    flowmat(:,300,a) = comp_in(:,1,a).* comLea(:,a);
    % Resulting composted fertilizer
    flowmat(:,315,a) = comp_in(:,1,a).* comFac(:,a);
end
% Resulting composted fertilizer - CORRECTING TOTAL MASS
% Substracting water and nutrient loss
flowmat(:,315,6) = flowmat(:,315,6) + sum(flowmat(:,298,2:nmat) + flowmat(:,300,2:nmat),3);
% check mass loss about 50%: flowmat(:,315,6)./ comp_in(:,1,6)
% Energy content calculated based on Channiwala 2002
% HHV = 0.3491*C + 1.1783*H + 0.1005*S - 0.1034*O - 0.015*N - 0.0211*Ash
% HHV in MJ/kg of dry mass
% Estimation for H S O Ash is based on their relationship to carbon content
% in feed. Based on Singh's work we know that H/C = 0.12, S/C = 0.026,
% and A/C = 0.68 (Ash content is larger for composted amnire ~60-70%
% compared to fresh manure ~20-30%. So we double of Singh's values,
% Ash/C = 1.4, see for example Matthiessen 2005. This is consistent
% with the ~51% reduction in Carbon content during composting that promotes
% a larger Ash/C ratio.
% Oxygen is calculated as the difference = 100 - ( C + H + N + S + Ash)
% check O/C = 0.95 approximately for poultry litter, UAn(:,4)./UAn(:,1)
UAn     = zeros(x,6);    % percentage [C H S O N Ash]
Drym    = flowmat(:,315,6) - flowmat(:,315,7);
UAn(:,1) = flowmat(:,315,4)./Drym * 100; % Check should be 10-50%
UAn(:,2:3) = UAn(:,1) * [0.12 0.026];
UAn(:,5) = flowmat(:,315,2)./Drym * 100;
UAn(:,6) = UAn(:,1) * 1.4;
UAn(:,4) = 100 - (sum(UAn(:,1:3),2) + sum(UAn(:,5:6),2));

%Energy in MJ/kg
comEner = 0.3491 * UAn(:,1) + 1.1783 * UAn(:,2) ...
       + 0.1005 * UAn(:,3) - 0.1034 * UAn(:,4)...
       - 0.0150 * UAn(:,5) - 0.0211 * UAn(:,6);

% Energy in kWh/y
flowmat(:,315,5) = Drym.* comEner * 1000 * 0.277778;

% Exported or imported organic fertilizer. Includes composted material and
% solids from food waste anaerobic digestion
% [used - produced]
flowmat(:,317,:) = flowmat(:,316,:) - flowmat(:,315,:) - flowmat(:,347,:);

%*****************************************************************
% EXPORTS/IMPORTS
%*****************************************************************
% Inorganic Fertilizer imported/exported (negative value is export)
% Includes land applied + algae growing - locally produced
flowmat(:,292,:) = flowmat(:,135,:) + FertAlg(:,1,:) - flowmat(:,293,:);
% Check if nutrient is being exported as a fertilizer. If so, the user must
% modify the distribution of fertilizer, i.e. increase the area of
% inorganic fertilizer applied in 'a_mix' row 1. We check for either
% nitrogen or phosphorus, depending which nutrient was defined as limiting
% by the user.
if any(flowmat(:,292,2:3) < 0)
    if edialog == 1;
    else
        emat(15) = 1;
    end
end
%*****************************************************************
```

```
% CORRECTING FLOWS AFTER WASTE MANAGEMENT IMPLEMENTATION
%*****************************************************************
% Soil infiltration of nutrients after leaching of landfill and composting
flowmat(:,83,:) = flowmat(:,83,:) + (flowmat(:,299,:) + flowmat(:,300,:));

% MODULE Energy sector
energys2;

ElecGen = sum(flowmat(:,260:263,5),2) + flowmat(:,257,5);
% Transmission and distribution losses
flowmat(:,343,5) = ElecGen * parause(173);
% EGeneration corrected by including transmission and distribution losses
flowmat(:,204,5) = ElecGen - flowmat(:,343,5);

% 205 Imports/Exports of Electricity
flowmat(:,205,:) = flowmat(:,195,:) - flowmat(:,204,:);

%*********************************************************************
% Estimation of characteristics of fuel for power generation
% (natural gas and diesel)
%*********************************************************************
% Coal is calculated in energy1.m
% It is assumed that biofuels (gas or liquids) are used first for power
% generation and later for local consumption (DO, CO, IN, TR)
% Natual Gas
flowmat(:,207,6) = (flowmat(:,207,5)./parause2(233)).* statin(47)/1000; % tonne/y
% Compostion depends on the mix of locally produced gas and imported gas
% Fraction of imported gas (energy balance)
impNG = (flowmat(:,207,5) - flowmat(:,278,5))./flowmat(:,207,5);

% flowmat(:,207,2:4) flowmat(:,278,2:4)
% Check carbon content
% flowmat(:,207,5)./flowmat(:,207,4)
% flowmat(:,278,5)./flowmat(:,278,4)
for a = 1:x
    if impNG(a) < 0 % all natural gas supplied by locally produced gas
    elseif impNG(a) > 0 % part of natural gas supplied by locally produced gas
        % In two attempts, imported and produced gas are added
        flowmat(a,207,2:4) = (flowmat(a,207,6) * (parause2(230:232)/100)').* impNG(a);
        flowmat(a,207,2:4) = flowmat(a,207,2:4) + flowmat(a,278,2:4);
    end
end
% Diesel
flowmat(:,208,6) = flowmat(:,208,5)./ parause2(245); % tonne/y
%
% Composition depends on the mix of locally produced and imported diesel
%
% Fraction of that is not supplied by local production (biofuel), therefore
% it has to be imported diesel (based on energy balance). This is
% calculated after the portion of fossil is substracted, parause(189).
impDI = (flowmat(:,208,5) * parause(189) - flowmat(:,264,5))./...
    (flowmat(:,208,5) * parause(189));
% Preventing NaN
impDI(isnan(impDI)) = 0;

% Fraction that is known to be fossil (and imported)
flowmat(:,235,5:6) = flowmat(:,208,5:6) * (1 - parause(189));
flowmat(:,235,2:4) = flowmat(:,235,6) * (parause2(242:244)/100)';

% flowmat(:,235,5)./flowmat(:,208,5)
% flowmat(:,208,2:4) flowmat(:,264,2:4)
```

```
for a = 1:x
    if impDI(a) < 0
        % diesel demand is surprassed by local production
        % In two attempts, imported and produced diesel are added
        flowmat(a,208,2:4) = flowmat(a,264,2:4) * (flowmat(a,208,5) *
parause(189)./flowmat(a,264,5));
        flowmat(a,208,2:4) = flowmat(a,208,2:4) + flowmat(a,235,2:4);
        % biodiesel use for power
        flowmat(a,248,:) = flowmat(a,264,:) * (flowmat(a,208,5) * parause(189)./flowmat(a,264,5));
    end
end
%*********************************************************************
% ESTIMATION OF FUEL REQUIREMENTS in physical units (tonne/y) for local
% consumption (non power)
% Includes calculation of imports or exports
%*********************************************************************
% Natural gas, tonnes/a
flowmat(:,196,6) = (flowmat(:,196,5)/parause2(233)).*statin(47)/1000;
% Compostion depends on the mix of locally produced gas and imported gas
% Remaining Natural gas
remNG = zeros(x,1,nmat);
% Composition of all local gas consumption as if it is supplied by fossil
allDfos = zeros(x,1,nmat); %
allDfos(:,:,2:4) = flowmat(:,196,6) * (parause2(230:232)/100)';

% flowmat(:,196,2:4) flowmat(:,278,2:4) flowmat(:,207,5)
for a = 1:x
    if  flowmat(a,196,6) == 0
        end

    % There is local consumption of natural gas
    else
        % Second IF term is just to verify that impNG is not NaN or Inf
        if impNG(a) < 0 && isnan(impNG(a) - impNG(a))
            end
         else % impNG(a) > 0 or produced natural gas has been used up for power
            % all gas is imported
            flowmat(a,196,2:4) = allDfos(a,:,2:4); % tonne/y
            % imported natural gas (fossil) INCLUDING FOR POWER
            flowmat(a,234,:) = flowmat(a,196,:) + flowmat(a,207,:) * impNG(a);
        end
    end
end
% Diesel (assumed mostly for transportation, so minimum and maximum apply)
flowmat(:,199,6) = flowmat(:,199,5)/parause2(245); % tonne/y
% Compostion depends on mix of locally produced diesel and imported diesel
% Remaining diesel
remDI = zeros(x,1,nmat);
% Composition of all local diesel consumption as if it is supplied by fossil
allDfos(:,:,2:4) = flowmat(:,199,6) * (parause2(242:244)/100)'; % tonne/y

% flowmat(:,196,2:4) flowmat(:,264,2:4) flowmat(:,208,5) flowmat(:,235,2:4)
% also flowmat(:,241,:), exported biofuel (liquid) is calculated
for a = 1:x

    % No local consumption of diesel
    if  flowmat(a,199,6) == 0
        end

    % There is local consumption of diesel
```

```
        else
            if impDI(a) < 0 && isnan(impDI(a))
                end
             else % impDI(a) > 0 or diesel has been used up for power
                % all diesel (fossil and bio) is imported
                flowmat(a,199,2:4) = allDfos(a,:,2:4); % tonne/y
                % imported diesel (fossil)
                flowmat(a,235,:) = flowmat(a,235,:) + flowmat(a,199,:) * (1 - parause(191));
                % imported diesel (bio)
                flowmat(a,241,:) = flowmat(a,199,:) * parause(191);
                flowmat(a,247,:) = flowmat(a,241,:);
            end
        end
    end
    % Gasoline characteristics and import/export estimation
    flowmat(:,200,6) = flowmat(:,200,5)/parause2(249); % tonne/y
    flowmat(:,200,2:4) = flowmat(:,200,6) * (parause2(246:248)/100)'; % tonne/y
    % 240   Imports/exports of gasoline (fossil fuel only and no production)
    % Assumes also that the minimum content of biofuels is achieved (imported)
    flowmat(:,240,:) = flowmat(:,200,:) * (1 - parause(191));
    % Import of bio-gasoline (includes biofuel for diesel engines)
    flowmat(:,241,:) = flowmat(:,241,:) + flowmat(:,200,:) * parause(191);
    % Total biofuel used for engines (gasoline and diesel)
    flowmat(:,247,:) = flowmat(:,247,:) + flowmat(:,200,:) * parause(191);

    %**********************************************************************
    % ESTIMATION OF EMISSIONS from local use and power
    %**********************************************************************
    % Emissions from fuel consumption in tonne/y (non power)
    emissionF       % by FUEL TYPE

    for a = 1:nmat
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
        flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
    % FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
        flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
    % SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
    % globally treated as part of biomass combustion emissions for commercial
    % or industrial use.
```

```
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
% FIREWOOD CALCULATIONS (emissions INCLUDE industrial biomass)
    flowmat(:,141,a) = -flowmat(:,123,a) * Combpro(a);
% SAWMILL RESIDUE CALCULATIONS (emissions) tonne/y. These emissions are
% globally treated as part of biomass combustion emissions for commercial
% or industrial use.
    flowmat(:,142,a) = -flowmat(:,120,a) * Combpro(a);
end
% Biomass emission factor tonne/kwh (assumed as wood)
bitkwh = parause2(129:131) * parause2(132)^-1;
bitkwh(2) = 0; % Phosphorus corrected to be zero

%%% Correction. Natural gas, gasoline, and diesel are assumed to
%%% undergo 100% oxidation, i.e. all C and N is released as emission
% Natural gas emission factor tonne/kwh (100% oxidation)
%ngtkwh = (parause2(230:232)/100) * (parause2(233)/statin(47))^-1 *0.001;
% Gasoline emission factor tonne/kwh (100% oxidation)
%gatkwh = (parause2(246:248)/100) * parause2(249)^-1;
% Diesel emission factor tonne/kwh (100% oxidation)
%ditkwh = (parause2(242:244)/100) * parause2(245)^-1;
% 219   Emissions from Local Coal use (DO, CO, IN, TR): ONLY IN
% Coal combustion generates CCP and emissions, partitioning input materials
tol_coal_em = zeros(x,3);
for a = 1:x
    tol_coal_em(a,1:3) = flowmat(a,198,2:4) - ccpPO(a,2,2:4);
end
flowmat(:,219,2:4) = -1 * (tol_coal_em);

% 220   Emissions from Local NG use (DO, CO, IN, TR)
% Includes biogas
flowmat(:,220,2:4) = -1 * flowmat(:,196,2:4);
% 221   Emissions from Local Diesel use (DO, CO, IN, TR)
% includes biodiesel
flowmat(:,221,2:4) = -1 * flowmat(:,199,2:4);
% 222   Emissions from Local Biomass use (DO, CO, IN, TR=0)
flowmat(:,222,2:4) = -1 * (sum(tol_biom,2) * bitkwh');

% 223   Emissions from Local Atl f1 use (DO, CO, IN, TR)
%flowmat(:,223,2:4) = -1 * (0);
% 224   Emissions from Local Atl f2 use (DO, CO, IN, TR)
%flowmat(:,224,2:4) = -1 * (0);
% 225   Emissions from Local Atl f3 use (DO, CO, IN, TR)
%flowmat(:,225,2:4) = -1 * (0);
% 226   Emissions from Local Gasoline use (DO, CO, IN, TR)
flowmat(:,226,2:4) = -1 * flowmat(:,200,2:4);
% 227   Emissions from Local Biodiesel use (DO, CO, IN, TR)
flowmat(:,227,2:4) = -1 * (0);

% 232   Total Emissions from Local Fuel Consumption
for a = 2:4
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
    flowmat(:,232,a) = sum(flowmat(:,219:227,a),2);
emissionU       % by USE
```

```
for a = 1:x
%[DO CO IN TR]
    % flowmat(:,228     Emissions from DO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,1)./ flowmat(a,196,5);
    flowmat(a,228,2:4) = -1 * (tut + tol_biom(a,1) * bitkwh);

    % flowmat(:,229 Emissions from CO fuel use (NG + Biomass):
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,2)./ flowmat(a,196,5);
    flowmat(a,229,2:4) =  -1 * (tut + tol_biom(a,2) * bitkwh);

    % flowmat(:,230     Emissions from IN fuel use (NG + Biomass + coal):
    % Coal from IN only, and calculated in emissionF.m
    tut(:,1) = flowmat(a,196,2:4).* tol_ngas(a,3)./ flowmat(a,196,5);
    flowmat(a,230,2:4) = -1 * (tut + tol_biom(a,3) * bitkwh + tol_coal_em(a,:)');

    % flowmat(:,231     Emissions from TR fuel use (NG + diesel + gasoline):
    flowmat(a,231,2:4) = -1 * (flowmat(a,196,2:4).* tol_ngas(a,4)./ flowmat(a,196,5)...
        + flowmat(a,199,2:4).* tol_dies(a,4)./ flowmat(a,199,5)...
        + flowmat(a,200,2:4).* tol_gaso(a,4)./ flowmat(a,200,5));

% Emissions in power generation applications. NG and diesel
% assumes 100% oxidation
flowmat(:,213,2:4) = -1 * flowmat(:,207,2:4);
flowmat(:,214,2:4) = -1 * flowmat(:,208,2:4);

%Altp1, Altp2, Altp3
%for a = 1:x
%    flowmat(a,215,2:4) = -1 * (flowmat(a,209,5) * ;
%    flowmat(a,216,2:4) = -1 * (flowmat(a,210,5) * ;
%    flowmat(a,217,2:4) = -1 * (flowmat(a,211,5) * ;
%end
% Total emissions from power generation
for a = 2:4
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
    flowmat(:,218,a) = sum(flowmat(:,212:217,a),2);
end
%*********************************************************************
% ESTIMATION OF environmental ENERGY (sun related)
%*********************************************************************
enviroE

QW = flowmat(:,83,5) + flowmat(:,295,5);

% Anthropogenic energy releases (usually INCOMING)
% {imported} Fuels + Electricity (from all sources)
QF = sum(flowmat(:,233:241,5),2) + sum(flowmat(:,261:263,5),2) + flowmat(:,205,5);

% Vaporized mass of water tonne/y [water forest grass crops impervious]
ME = [Et_vol(:,1) times(Et_vol(:,2:4),smfactor) -flowmat(:,78,1)] * 1382000;
% Vaporization energy kWh/y (OUTGOING)
QEforH = ME * wl * 1.1622;
% We prefer using all evaporation energy (including industrial and power)
QE = flowmat(:,87,5);

% alpha parameter for each land terrain type (Grimmond 2002) as a function
% of covered area
% [water forest grass crops impervious]
Palpha = ones(x,1) * [0.8 0.8 0.8 0.8 0.2];
% Pbeta = 3 W/m2 for urban, 20 W/m2 for rural -- converted to kWh/y
```

```
Pbeta = (ones(x,1) * [20 20 20 20 3]) * 365 * 24 * (1000000/1000).* Et_area ;

% Turbulent sensible heat flux kWh/y (OUTGOING)
% [water forest grass crops impervious]
QH = ((1./Palpha).*(1 + 1/pp) - 1).* (QEforH - Pbeta) - Pbeta;
flowmat(:,313,5) = -1 * sum(QH,2);

% Net radiation in kWh/y (note: Q_Et in cal/cm2/day) (INCOMING)
% [water forest grass crops impervious]
Qstar = (ones(x,1) * (Q_Et(:,2)' * 10^10 * 365 *  1.1622 / 1000000)).* Et_area;
% Effective Solar Radiation Input kWh/y
flowmat(:,283,5) = sum(Qstar,2);

% Storage heat flux kWh/y (heat emitted from buildings and land)
% QS = Q* + QF - QE - QH - QW
QS = sum(Qstar,2) + QF + QE - sum(QH,2) + QW;

flowmat(:,314,5) = -1 * QS;
% QS./Qstar
% flowmat(:,314,5) ./flowmat(:,283,5)
% Annual average for the city of Atlanta: 4.37 kWh/m2/day
% Source: Whitlock, C. E., et al., Release 3 NASA Surface Meteorology and
% Solar Energy Data Set for Renewable Energy Industry Use.
% Rise & Shine 2000, the 26th Annual Conference of the Solar
% Energy Society of Canada Inc. and Solar, Oct. 21-24, 2000,
% Halifax, Nova Scotia, Canada.
% Compare to 4.37 kWh/m2/day
SolarR = Q_Et(:,2) * (1.1622/1000000) * 10000;

% Converts energy and water units
conversion

flowmat(:,:,5) = flowmat(:,:,5)./ 1000000;
% Convert back flowmat(:,:,5) = flowmat(:,:,5).* 1000000;
%************************************************************************
% CONVERTS ALL WATER FLOWS FROM MGD to m3/s
%************************************************************************
flowmat(:,:,1) = flowmat(:,:,1).*(1000000 * 0.0037854 / (24 * 3600));

% MODULE generate indicators and generates output tables
out_main;

healthyE

Dfix = zeros(x,1);
for a = 1:x
    Dfix(a,1)= Drain(1,1,a)/AbaseD;
end
%************************************************************************
% Water
%************************************************************************
% * Air emission precipitation - evapotranspiration.
% * Aquatic emission (runoff + leaching), same magnitude - different sign
ae0(:,7) = flowmat(:,81,7) ...
    - ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1));

we0(:,7) = ((land * Q_Et(2,4) * 0.00072376 * 1382000).* smfactor(:,1))...
     - flowmat(:,81,7);

%************************************************************************
```

```
% Nitrogen
%************************************************************************
% Total nitrogen flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition wet
% + deposition dry
% + fixation
% - denitrification
ae0(:,2) = parause2(41) * flowmat(:,81,1) * 1.382 + ...
    land* 100 *(parause2(45) + parause(125) - parause(128))/1000;

% Runoff and infiltration flows are estimated from the remaining water (not
% evaporated) and the infiltration rate of pervious surfaces.
% Load estimation: runoff + infiltration (all forest)
ru = -1 * parause2(25) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(141) * parause2(262) * land.* Dfix;
we0(:,2) = sum([ru in],2);

%************************************************************************
% Phosphorus
%************************************************************************
% Total phosphorus flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + deposition (wet and dry)
ae0(:,3) = parause2(42) * flowmat(:,81,1) * 1.382 + land* 100 * parause2(46)/1000;

% Runoff and infiltration loads: runoff + infiltration (all forest)
ru = -1 * parause2(26) * 100/1000 * land * inout(:,1)/statin(27);
in = -1 * parause(142) * parause2(263) * land.* Dfix;
we0(:,3) = sum([ru in],2);

%************************************************************************
% Carbon
%************************************************************************
% Total carbon flux (all values in tonnes/y) assumes all area is forest
% and is estimated based on:
% + photosynthesis/respiration flux (adjusted to total area) + deposition
% - metabolic respiration
% Change in biomass (above and below ground)
fbiomass = noTgrowth * (1 + parause(123)) * parause2(131).* (land./(noTarea/100));
% Dead organic mass due to mortality
fdom = fbiomass * parause(120);
% Change in organic carbon in mineral soils
DCminForest = zeros(x,1);
for a = 1:x-1
% ANNUAL CHANGE IN CARBON STOCKS IN SOILS
end
Dsoilc = DCminForest - parause(102) * land * ones(x,1);
% Annual C flux due to photosynthesis and respiration
PhotoFlux = fbiomass  + fdom + Dsoilc;
% Annual C flux due to metabolic respiration (this would depend on animal
% population within the area)
MetabFlux = 0;
% Total C flux
% + photosynthesis and respiration (also metabolic)
% + deposition (wet and dry)
ae0(:,4) = PhotoFlux + parause2(43) * flowmat(:,81,1) * 1.382 + ...
    land* 100 * parause2(47)/1000 - MetabFlux;

% Runoff and infiltration loads: runoff + infiltration (all forest)
% runoff is calculated differently from other nutrients by using the
```

```
% organic matter (in forest floor) and the runoff factor.
ru = -1 * parause2(264) * parause(139).* land.* inout(:,1)/statin(27);
in = -1 * parause2(264) * parause(140).* land.* Dfix;
we0(:,4) = sum([ru in],2);

%************************************************************************
% Energy - in GWh/y
%************************************************************************
% Energy balance based on:
% QS = Q* + QF - QE - QH - QW
% Net wave radiation + Anthropogenic releases =
% vaporization + sensible heat flux + heat storage flux
% no QF because 'forest' is the reference state
% All forest evapotranspiration kWh/y
ME_F = land.* Q_Et(2,4) * 0.00072376 * 1382000 * smfactor(:,1);
QE_F = ME_F * wl * 1.1622;

% ALL FOREST Turbulent sensible heat flux kWh/y (OUTGOING)
QH_F = ((1./Palpha(:,2)).*(1 + 1/pp) - 1).* (QE_F - Pbeta(:,2)) - Pbeta(:,2);

% All forest net radiation kWh/y
Qstar_F = (Q_Et(2,2)' * 10^10 * 365 *  1.1622 / 1000000).* land;

%------------------------------------------------------------------------
% It is assumed that infiltrated water has a 5 C temperature difference
% with the equilibrium temperature, so there is some thermal energy
% available
we0(:,5)= we0(:,7) * 5 * 1.1622/1000000; %GWh/y
%------------------------------------------------------------------------
% All forest Storage Heat Flux kWh/y
QS_F = Qstar_F  - QE_F - QH_F + we0(:,5) * 1000000;

% Energy emission from reference state (healthy emission)
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
% - Losses in water (evaporated and surface)
ae0(:,5) = Qstar_F - (QH_F + QS_F + QE_F - we0(:,5) * 1000000);
ae0(:,5) = ae0(:,5)/1000000; % GWh/y. 5 is added to avoid zero

%************************************************************************
% Rest - no healthy emissions defined
%************************************************************************
ae0(:,1) = -1; % ficticious value to run model
we0(:,1) = -1; % ficticious value to run model
ae0(:,6) = -1; % ficticious value to run model
we0(:,6) = -1; % ficticious value to run model

%*********************************************************************
% CATEGORIZATION OF FLOWS FOR EACH SECTOR % (year, material, sector)
%*********************************************************************
% Flows in each sector are classified as resource, product, waste, aquatic
% emission, and air emission. Material and energy storage is also
% calculated
flowClass

I284 = ones(x,1,nmat);
I284(flowmat(:,284,:)<0) = 0;

I285 = ones(x,1,nmat);
```

```
I285(flowmat(:,285,:)<0) = 0;

pro_vec(:,:,1) = flowmat(:,93,:) + flowmat(:,284,:).* (1-I284) ...
    + flowmat(:,285,:).* (1-I285);
res_vec(:,:,1) = flowmat(:,3 ,:) + flowmat(:,284,:).* (I284) ...
    + flowmat(:,285,:).* (I285);
was_vec(:,:,1) = 0;
% air emissions: total evaporation + wwtp emissions + precipitation +
% septic tank emissions
ae1_vec(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,81,:) + flowmat(:,318,:);
ae1_out(:,:,1) = flowmat(:,87,:) + flowmat(:,130,:) + flowmat(:,318,:);
% water emissions: soil infiltration and surface runoff
%(used in enviroE for QW)
we1_vec(:,:,1) = flowmat(:,83,:) + flowmat(:,295,:) + flowmat(:,331,:);

% Accumulation
store(:,:,1) = pro_vec(:,:,1) + res_vec(:,:,1) - was_vec(:,:,1) + ...
    ae1_vec(:,:,1) + we1_vec(:,:,1);

%**********************************************************************
% Forestry Sector
%**********************************************************************
% (year, material, sector)
pro_vec(:,:,2) = flowmat(:,124,:);
res_vec(:,:,2) = flowmat(:,106,:);
was_vec(:,:,2) = 0;
% denitrification wwtp + N fixation + N volatilization + deposition +
% photosynthesis + Metabolic Respiration + firewood combustion
ae1_vec(:,:,2) = flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:)...
    + flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:) + ...
    flowmat(:,141,:);
ae1_out(:,:,2) = flowmat(:,146,:) + flowmat(:,140,:)...
    + flowmat(:,148,:) + flowmat(:,147,:) + flowmat(:,141,:);
we1_vec(:,:,2) = 0;
% Accumulation in the wood local market
dWdt = (flowmat(:,106,:) - flowmat(:,101,:) + flowmat(:,100,:)) + ...
    flowmat(:,103,:) + flowmat(:,121,:) - flowmat(:,116,:) - ...
    flowmat(:,118,:) + flowmat(:,114,:) + flowmat(:,141,:);

flowmat(:,250,2:5) = dWdt(:,:,2:5);

% Accumulation in soils (forests and crops)
dFdt = (flowmat(:,146,:) + flowmat(:,145,:) + flowmat(:,140,:) + ...
    flowmat(:,134,:) + flowmat(:,148,:) + flowmat(:,147,:))...
    - flowmat(:,97,:) + (flowmat(:,98,:) - flowmat(:,144,:)) ...
    - flowmat(:,119,:) + flowmat(:,139,:) - ...
    (flowmat(:,177,:) + flowmat(:,164,:)); % these last two: feed + crops

flowmat(:,249,2:5) = dFdt(:,:,2:5);

% Total Annual Accumulation
store(:,:,2)  = pro_vec(:,:,2) + res_vec(:,:,2) - was_vec(:,:,2) + ...
    ae1_vec(:,:,2) + we1_vec(:,:,2);

%**********************************************************************
% Food Sector
%**********************************************************************
 % If export (PRODUCT) flow negative
I178 = ones(x,1,nmat);
I178(flowmat(:,178,:)<0) = 0;
```

```
I172 = ones(x,1,nmat);
I172(flowmat(:,172,:)<0) = 0;

ILi = ones(x,1,nmat);
ILi(sum(flowmat(:,192:194,:),2)<0) = 0;

% (year, material, sector)
pro_vec(:,:,3) = flowmat(:,178,:).* (1-I178) + flowmat(:,172,:).* (1-I172)...
    + sum(flowmat(:,192:194,:),2).* (1-ILi);
res_vec(:,:,3) = flowmat(:,178,:).* I178 + flowmat(:,172,:).* I172 + ...
    sum(flowmat(:,192:194,:),2).* ILi + flowmat(:,187,:);
was_vec(:,:,3) = 0;
% Air emissions from livestock operations. Air Emissions from cropland are
% included in forestry
ae1_vec(:,:,3) = flowmat(:,294,:);
ae1_out(:,:,3) = flowmat(:,294,:);
we1_vec(:,:,3) = 0;

% Annual accumulation in the livestock industry
dLdt = flowmat(:,176,:) + sum(flowmat(:,192:194,:),2) + flowmat(:,294,:)...
    - flowmat(:,182,:)- flowmat(:,160,:) - flowmat(:,162,:) +...
    flowmat(:,306,:);

% Annual Accumulation in the food Local market
dFdt = flowmat(:,156,:) - flowmat(:,117,:);

% Annual accumulation FOOD SECTOR
flowmat(:,251,2:5) = dFdt(:,:,2:5) + dLdt(:,:,2:5);    % For the record :)

% Apparent accumulation (for calculating indicators)
store(:,:,3)   = pro_vec(:,:,3) + res_vec(:,:,3) - was_vec(:,:,3) +...
    ae1_vec(:,:,3) + we1_vec(:,:,3);

%**************************************************************************
% Energy Sector
%**************************************************************************
 % If export (PRODUCT) flow negative
Iene = ones(x,1,nmat);
Iene(sum(flowmat(:,233:241,:),2)<0) = 0;

Iele = ones(x,1,nmat);
Iele(flowmat(:,205,:)<0) = 0;

% [year, material, sector]
pro_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* (1 - Iene) + ...
    flowmat(:,205,:).* (1 - Iele);
res_vec(:,:,4) = sum(flowmat(:,233:241,:),2).* Iene + ...
    flowmat(:,205,:).* Iele + flowmat(:,246,:);
was_vec(:,:,4) = 0;
% - emissions local consumption
% - emission biofuel production
% - emissions power genereneration
% + net all-wave radiation input
% - turbulent sensible heat flux
% - storage het flux
ae1_vec(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
    + flowmat(:,283,:) + flowmat(:,313,:) + flowmat(:,314,:);

% ONLY OUTGOING FLUXES (for calculation of eco-efficiency indicator)
ae1_out(:,:,4) = flowmat(:,232,:) + flowmat(:,243,:) + flowmat(:,218,:)...
```

```
    + flowmat(:,313,:) + (flowmat(:,314,:)>0).* flowmat(:,314,:);
we1_vec(:,:,4) = 0;
% Total Annual Accumulation
store(:,:,4) = pro_vec(:,:,4) + res_vec(:,:,4) - was_vec(:,:,4) + ...
    ae1_vec(:,:,4) + we1_vec(:,:,4);

%**************************************************************************
% Waste Management Sector
%**************************************************************************
 % If export (PRODUCT) flow negative
Ifer = ones(x,1,nmat);
Ifer(flowmat(:,292,:)<0) = 0;

% (year, material, sector)
pro_vec(:,:,5) = flowmat(:,112,:) + flowmat(:,292,:).*(1 - Ifer) + ...
    flowmat(:,291,:) + flowmat(:,312,:) + flowmat(:,342,:);
res_vec(:,:,5) = flowmat(:,292,:).* Ifer;

% Remaining Landfilled material after air emissions and leaching
was_vec(:,:,5) = flowmat(:,301,:);
% air emissions =
% Air emissions released from Landfill
% Emissions from wood products (PWY) in MSW Incineration
% Emissions from FOOD products in MSW incineration
% Emission from biomass composting
% Emission from the struvite process
ae1_vec(:,:,5) = flowmat(:,272,:) + flowmat(:,143,:) + flowmat(:,280,:)+ flowmat(:,298,:) +
flowmat(:,341,:);
ae1_out(:,:,5) = ae1_vec(:,:,5);
% water emissions
we1_vec(:,:,5) = flowmat(:,299,:) + flowmat(:,300,:);

% Total Annual Accumulation
store(:,:,5) = pro_vec(:,:,5) + res_vec(:,:,5) - was_vec(:,:,5) + ...
    ae1_vec(:,:,5) + we1_vec(:,:,5);

%**************************************************************************
% INDICATOR SCORES ARE CALCULATED                                         *
%**************************************************************************
% indimat = zeros(x,nmat,nindi);
% As a reference, a list of indicators and their indexes is presented:
%       1       Product index
%       2       Waste Index
%       3       Waste Eco-efficiency
%       4       Emission Eco-efficiency
%       5       Health of air Emissions
%       6       Health of aquatic Emissions
%       7       Waste equals food index
%       8       "available"
%       9       new
% The matrix of indicators is generated based on the categorization
% performed above. (years, substance, indicator index)
indimat(:,:,1) = abs(sum(pro_vec,3)./ sum(res_vec,3));
indimat(:,:,2) = abs(sum(res_vec,3)./ sum(was_vec,3));
indimat(:,:,3) = abs(sum(pro_vec,3)./ sum(was_vec,3));
%indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_vec + we1_vec),3));
% EEI calculated only with outgoing air emissions, using 'ae1_out'
% (and water emissions are always outgoing).
indimat(:,:,4) = abs(sum(pro_vec,3)./sum((ae1_out + we1_vec),3));
% '5' is added to both numerator and denominator to avoid zero,
% particularly for 'ae0' which is usually zero for forests (reference
```

```
% state)
indimat(:,:,5) = (ae0 + 5)./(sum(ae1_vec,3) + 5);
indimat(:,:,6) = we0./sum(we1_vec,3);
indimat(:,:,7) = -sum(pro_vec,3)./(sum(res_vec,3) - sum(store,3) + ae0 + we0);
indimat(:,:,8)  = 0;
%indimat(:,:,8)  = abs((Ao.^alpha).*pal1 + (Bo.^beta).*pbe1 + (Go.^gamma).*pga1);
%************************************************************************
%% NEW indicator
%************************************************************************
% efficiencies
%167    Coal power plant energy efficiency
%168    NG power plant energy efficiency
%169    Diesel power plant energy efficiency
%205    Internal combustion efficiency for engines gasoline or diesel
%206    Internal combustion efficiency for engines Natural Gas  ratio
%207    Household and industrial firewood combustion
%208    Household and industrial natural gas combustion
effic = [
(0.32)
(0.55)
(0.38)
(0.25)
(0.9)
(0.4)
(0.65)];
%% Energy
%(sectoral scope)
%performance in the water sector (incl technologies)
Temp = flowmat(:,332,5) + flowmat(:,333,5) + ...
    flowmat(:,334,5) + flowmat(:,338,5) + ...
    flowmat(:,339,5) + flowmat(:,340,5);
E_dem_ws1 = Temp./effic(2);
% Energy (primary) generated, GWh/a
Temp = flowmat(:,265,5) * effic(3) + flowmat(:,266,5) * effic(2) + ...
    flowmat(:,277,5)*effic(2) + flowmat(:,304,5)*effic(3) +...
    flowmat(:,335,5,:)*0.3;
E_gen_ws = Temp./effic(2);
% Energy indicator Water sector
% Demand converted into natural gas terms based on efficiency
%As = E_gen_ws./E_dem_ws1;
% (systems scope)
% Calculation of all energy demand for the whole system in natural gas
% terms (using efficiency)
Temp = flowmat(:,205,5) + flowmat(:,196,5) * 0.5 * effic(5) + ...
    flowmat(:,196,5) * 0.5 * effic(7) + ...
    flowmat(:,233,5) * effic(1) + flowmat(:,207,5) * effic(2) + ...
    flowmat(:,236,5) * effic(6) + (flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5)) * effic(4);
E_dem_ws2 = Temp./effic(2);
% Energy indicator whithin whole system
% Demand converted into natural gas terms based on efficiency
As = E_gen_ws./E_dem_ws2;
indimat(:,5,9) = As;

%% Water
% Public supply
indimat(:,7,9) = flowmat(:,6,1);
% All water withdrawals
%indimat(:,7,9) = flowmat(:,3,1);
```

```
%% Nitrogen
%(sectoral scope)
% N recovery from food
Temp = flowmat(:,136,2) + flowmat(:,293,2) + flowmat(:,342,2);
N_rec_ws = Temp;
Food = flowmat(:,156,2);
%As = N_rec_ws./Food;
% (systems scope)
% N recovery from inputs: wood, industrial wastewater, food
Ninput = flowmat(:,156,2) + flowmat(:,90,2) +...
    flowmat(:,106,2) + flowmat(:,100,2) + flowmat(:,99,2);
As = N_rec_ws./Ninput;
indimat(:,2,9) = As;

%% Phosphorus
% P recovery from food
Temp = flowmat(:,136,3) + flowmat(:,293,3) + flowmat(:,342,3);
P_rec_ws = Temp;
Food = flowmat(:,156,3);
As = P_rec_ws./Food;
% P recovery from inputs: wood, industrial wastewater, food
Pinput = flowmat(:,156,3) + flowmat(:,90,3) +...
    flowmat(:,106,3) + flowmat(:,100,3) + flowmat(:,99,3);
%As = P_rec_ws./Pinput;
indimat(:,3,9) = As;

%% Carbon
% Carbon ratio with the water sector demand (natural gas terms)
% Demand
% Conversion to tonnes of carbon assuming:
% - natural gas as the energy source
% - same proportion of carbon and energy as for power generation
r_NG_Energy = flowmat(:,207,4)./flowmat(:,207,5);
E_dem_wsC1 = E_dem_ws1.* r_NG_Energy;
% Generation of fuels
Temp = sum(flowmat(:,265:266,4),2) + flowmat(:,277,4) + ...
    flowmat(:,304,4) + flowmat(:,335,4);
C_emm_ws = Temp;
As = C_emm_ws./E_dem_wsC1;
% Carbon ratio with the whole systems demand (natural gas terms)
% Demand of the whole system
Temp = flowmat(:,205,5) + flowmat(:,196,5) + ...
    flowmat(:,196,5) + ...
    flowmat(:,233,5) + flowmat(:,207,5)  + ...
    flowmat(:,236,5) + flowmat(:,235,5) + ...
    flowmat(:,240,5) + flowmat(:,241,5) + ...
    flowmat(:,247,5);
E_dem_wsC2 = Temp.* r_NG_Energy;
%As = C_emm_ws./E_dem_wsC2;
indimat(:,4,9) = As;

      % stores values for each realization
      infbodRSA(:,sa)  = infBOD;
      effbodRSA(:,sa)  = effBOD;
      indimatRSA(:,:,:,sa)  = indimat;
      flowmatRSA(:,:,:,sa)  = flowmat;
      errorsRSA(:,sa)       = emat;
      statoutRSA(:,:,sa)    = statout;
      % Values for healthy emissions
      airRSA(:,:,sa) = ae0;
      watRSA(:,:,sa) = we0;
```

```
      mflowRSA(:,:,1,sa) = sum(pro_vec,3);
      mflowRSA(:,:,2,sa) = sum(res_vec,3);
      mflowRSA(:,:,3,sa) = sum(was_vec,3);
      mflowRSA(:,:,4,sa) = sum(ae1_vec,3);
      mflowRSA(:,:,5,sa) = sum(we1_vec,3);

      % We compare those outputs to the behavior definition.
      behdef1;

nbeh = zeros(x,1);
bT = zeros(x,1);

if modset(19) > 0

else
% base_v = Mean value of indicators, base case, time series
% base_ID = identification # of indicators: [#, # indicator, species]
% Both matrices loaded in dataready; retrieved from MSSAout.xls
% modset(17) is the indicator of interest;
% modset(18) is the degree of improvement (as a percentage);
% Matrix of objective value for indicator of interest, obje_v
% Actual values of the indicator of interest (time series)
indi_value = indimat(:,base_ID(modset(17),3),base_ID(modset(17),2));

% Degree of improvement (0 to 1). Assumed to be the same for every year.
% The user wants an automated calculation of the improvement
obje_v = zeros(size(base_v));
% Test of behavior or non-behavior(0 or 1)
% If one value of the time series of the indicator of interest exceeds the
% objective value then we have found a BEHAVIOR
if modset(18) ~= 0
    %end
else
% The user wants to define the improvement explicitly (in input file) MSA
% bring the value input by the user through 'dataready.m'
    obje_v = obje_v2;

    % user
    % 1 if the user wants to define the model output manually (the one
    % which is compared to the objective or goal.
    % 0 if an indicator is used as the model ouput of interest
    meL = 0;
    if meL > 0
    else
        nbeh(indi_value < obje_v & obje_v < base_v) = 1;
        nbeh(indi_value < obje_v & obje_v > base_v) = 0;
        nbeh(indi_value > obje_v & obje_v > base_v) = 1;
        nbeh(indi_value > obje_v & obje_v < base_v) = 0;
    end
end
% Behavior / non-behavior results
bT = nbeh;

      beh(:,sa) = bT;

end
%%
% Split 'norm' into regular parameters and 'inout' parameters
% Identify how many 'inout' parameters are part of the RSA
```

```
numIO = sum(input(size(input,1)-2,:));
% Rearrange parameters associated with 'inout'
x_norm = norm(:,1:size(norm,2) - numIO*x);
i_norm = norm(:,size(norm,2) - numIO*x + 1:size(norm,2));
i_norm = permute(i_norm,[2 1]);
% Add 'years' as a dimension
reg_norm = zeros(x,size(x_norm,1),size(x_norm,2));
ino_norm = zeros(x,nsim,numIO);
% Building final matrices
for yr = 1:x
    reg_norm(yr,:,:) = x_norm;
end
for pa1 = 1:numIO
    ino_norm(:,:,pa1) = i_norm(pa1*x-x+1:pa1*x,:);
    ino_norm(:,:,pa1) = i_norm(pa1*x-x+1:pa1*x,:);
end
% Parameters concatenated again (adjusted by 'inout')
% format [years, nsim, npar]
norm2 = cat(3,reg_norm,ino_norm);
%%
% Split 'abso' into regular parameters and 'inout' parameters
% Identify how many 'inout' parameters are part of the RSA
numIO = sum(input(size(input,1)-2,:));
% Rearrange parameters associated with 'inout'
x_abso = abso(:,1:size(abso,2) - numIO*x);
i_abso = abso(:,size(abso,2) - numIO*x + 1:size(abso,2));
i_abso = permute(i_abso,[2 1]);
% Add 'years' as a dimension
reg_abso = zeros(x,size(x_abso,1),size(x_abso,2));
ino_abso = zeros(x,nsim,numIO);
% Building final matrices
for yr = 1:x
    reg_abso(yr,:,:) = x_abso;
end
for pa1 = 1:numIO
    ino_abso(:,:,pa1) = i_abso(pa1*x-x+1:pa1*x,:);
    ino_abso(:,:,pa1) = i_abso(pa1*x-x+1:pa1*x,:);
end
% Parameters concatenated again (adjusted by 'inout')
% format [years, nsim, npar]
abso2 = cat(3,reg_abso,ino_abso);
%%
% Number of parameters redefined (adjusted by 'inout')
npar = size(norm2,3);
% Define matrix for sensitivity class for each year
% Define matrix for z- or d-statistics for each year
% Define simple probability matrix for each year
% Define matrix for abs values of all parameters, segregated by B and NB
seg_abso = zeros(nsim,npar,x);

sc_y = zeros(npar,x);
zd_y = zeros(npar,x);
pr_y = zeros(npar,x);

% separate behaviour (b=1) and nonbehaviour (b=0) matrixes and
% compute the Kolmogorov-Smirnov d-statistic, and determine the
% sensitivity class for each of the sampled model parameters
for yr = 1:x
% permuted format [nsim, npar, years]
a0n = permute(norm2(yr,beh(yr,:)==0,:),[2 3 1]);
a1n = permute(norm2(yr,beh(yr,:)==1,:),[2 3 1]);
```

```
a0n_abso = permute(abso2(yr,beh(yr,:)==0,:),[2 3 1]);
a1n_abso = permute(abso2(yr,beh(yr,:)==1,:),[2 3 1]);
%size(a1n_abso) abso2)
% 1.- We plot b and non-b curves for the parameter and year of interest
if modset(20) > 0 && yr == modset(10)
    % first plot
    figure('name',strcat('Parameter ',num2str(modset(20))));
     plot_c(a0n(:,modset(20)),1); % plot the generation of d
     hold on;
     plot_c(a1n(:,modset(20)),2);%
     hold off;
end
n0 = size(a0n,1);
n1 = size(a1n,1);
n01 = sqrt((n0 + n1)/(n0*n1));

% z = d / n01
% use d- or z-statistics? z-statistics is independent of the number of
% behavior and non-behavior that result from the RSA simulation
% convert to z-statistic (see Osidele 2006 Osidele 2006 Application of
% RSA to a Performance Assessment Model of high level repository IHLRWM)
% user
usezd = 1; % 1 for z and 0 for d
if usezd < 1
else
n02 = sqrt((n0 + n1)/(n0*n1));
end
% d-statistic matrix [0.1% 1% 5% 10% 15% 20%] significance
czd = [1.95 1.63 1.36  1.22 1.14 1.07]' * n01/n02;
% define bin centres
bc = [0.05:0.1:0.95];

% define z- and d-statistic (zd) matrix and sensitivity class (sc) matrix
zd = zeros(npar,1);
sc = zeros(npar,1);

for j=1:npar % for each parameter

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);

   % Calculation of z- or d-statistic
   zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

   % determine the sensitivity class (sc)
   if zd(j) > czd(3)
   elseif zd(j) < czd(5)
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);
```

```
bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
```

```
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);

   % Calculation of z- or d-statistic
   zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

   % determine the sensitivity class (sc)
   if zd(j) > czd(3)
   elseif zd(j) < czd(5)
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);

   % Calculation of z- or d-statistic
   zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

   % determine the sensitivity class (sc)
   if zd(j) > czd(3)
   elseif zd(j) < czd(5)
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);

   % Calculation of z- or d-statistic
   zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

   % determine the sensitivity class (sc)
   if zd(j) > czd(3)
   elseif zd(j) < czd(5)
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
```

```
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;
```

```
% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);
```

```
    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
```

```
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;
```

```
% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);
```

```
        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
```

```
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
```

```
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
```

```
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);

   % Calculation of z- or d-statistic
   zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

   % determine the sensitivity class (sc)
   if zd(j) > czd(3)
   elseif zd(j) < czd(5)
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);

   % Calculation of z- or d-statistic
   zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

   % determine the sensitivity class (sc)
   if zd(j) > czd(3)
   elseif zd(j) < czd(5)
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);

   % Calculation of z- or d-statistic
   zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

   % determine the sensitivity class (sc)
   if zd(j) > czd(3)
   elseif zd(j) < czd(5)
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);
```

```
% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;
```

```
% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
```

```
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
```

```
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;
```

```
% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);
```

```
bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
```

```
    else
       sc(j) = 2;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
```

```
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;
```

```
% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);
```

```
% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
```

```
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
```

```
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
```

```
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;
```

```
% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);
```

```
bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
```

```
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
```

```
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;
```

```
% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);
```

```
% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
```

```
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;
```

```
    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
```

```
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
```

```
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);
```

```
        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
```

```
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
```

```
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;
```

```
% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);
```

```
% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
```

```
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;
```

```
% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);
```

```
bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
```

```
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
```

```
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
```

```
         sc(j) = 3;

      % compute the d-statistic
      bin0 = hist(a0n(:,j),bc);
      bin0 = bin0/n0;
      bin0 = cumsum(bin0);

      bin1 = hist(a1n(:,j),bc);
      bin1 = bin1/n1;
      bin1 = cumsum(bin1);

      % Calculation of z- or d-statistic
      zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

      % determine the sensitivity class (sc)
      if zd(j) > czd(3)
      elseif zd(j) < czd(5)
         sc(j) = 3;

      % compute the d-statistic
      bin0 = hist(a0n(:,j),bc);
      bin0 = bin0/n0;
      bin0 = cumsum(bin0);

      bin1 = hist(a1n(:,j),bc);
      bin1 = bin1/n1;
      bin1 = cumsum(bin1);

      % Calculation of z- or d-statistic
      zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

      % determine the sensitivity class (sc)
      if zd(j) > czd(3)
      elseif zd(j) < czd(5)
         sc(j) = 3;

      % compute the d-statistic
      bin0 = hist(a0n(:,j),bc);
      bin0 = bin0/n0;
      bin0 = cumsum(bin0);

      bin1 = hist(a1n(:,j),bc);
      bin1 = bin1/n1;
      bin1 = cumsum(bin1);

      % Calculation of z- or d-statistic
      zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

      % determine the sensitivity class (sc)
      if zd(j) > czd(3)
      elseif zd(j) < czd(5)
         sc(j) = 3;

      % compute the d-statistic
      bin0 = hist(a0n(:,j),bc);
      bin0 = bin0/n0;
      bin0 = cumsum(bin0);

      bin1 = hist(a1n(:,j),bc);
      bin1 = bin1/n1;
      bin1 = cumsum(bin1);
```

```
% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;
```

```
% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
```

```
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
```

```
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;
```

```
% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);
```

```
bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
```

```
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
```

```
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;
```

```
% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);
```

```
% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
```

```
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
```

```
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
```

```
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;
```

```
% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);
```

```
        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
```

```
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    else
       sc(j) = 2;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
```

```
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;
```

```
% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);
```

```
% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
```

```
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;
```

```
% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
```

```
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
```

```
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);
```

```
        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
```

```
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);

   % Calculation of z- or d-statistic
   zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

   % determine the sensitivity class (sc)
   if zd(j) > czd(3)
   elseif zd(j) < czd(5)
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);

   % Calculation of z- or d-statistic
   zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

   % determine the sensitivity class (sc)
   if zd(j) > czd(3)
   elseif zd(j) < czd(5)
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);

   % Calculation of z- or d-statistic
   zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

   % determine the sensitivity class (sc)
   if zd(j) > czd(3)
   elseif zd(j) < czd(5)
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
```

```
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;
```

```
% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);
```

```
% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
```

```
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;
```

```
% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);
```

```
bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
```

```
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
```

```
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
```

```
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);

   % Calculation of z- or d-statistic
   zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

   % determine the sensitivity class (sc)
   if zd(j) > czd(3)
   elseif zd(j) < czd(5)
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);

   % Calculation of z- or d-statistic
   zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

   % determine the sensitivity class (sc)
   if zd(j) > czd(3)
   elseif zd(j) < czd(5)
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);

   % Calculation of z- or d-statistic
   zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

   % determine the sensitivity class (sc)
   if zd(j) > czd(3)
   elseif zd(j) < czd(5)
      sc(j) = 3;

   % compute the d-statistic
   bin0 = hist(a0n(:,j),bc);
   bin0 = bin0/n0;
   bin0 = cumsum(bin0);

   bin1 = hist(a1n(:,j),bc);
   bin1 = bin1/n1;
   bin1 = cumsum(bin1);
```

```
% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;
```

```
% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
```

```
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
```

```
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;
```

```
% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);
```

```
        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
```

```
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
```

```
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;
```

```
% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);
```

```
% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
```

```
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
```

```
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
```

```
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;
```

```
% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);
```

```
bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
```

```
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
```

```
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;
```

```
% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);
```

```
% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
```

```
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

        % determine the sensitivity class (sc)
        if zd(j) > czd(3)
        elseif zd(j) < czd(5)
           sc(j) = 3;

        % compute the d-statistic
        bin0 = hist(a0n(:,j),bc);
        bin0 = bin0/n0;
        bin0 = cumsum(bin0);

        bin1 = hist(a1n(:,j),bc);
        bin1 = bin1/n1;
        bin1 = cumsum(bin1);

        % Calculation of z- or d-statistic
        zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;
```

```
% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
```

```
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
```

```
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
else
   sc(j) = 2;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);
```

```
bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
   sc(j) = 3;

% compute the d-statistic
bin0 = hist(a0n(:,j),bc);
bin0 = bin0/n0;
bin0 = cumsum(bin0);

bin1 = hist(a1n(:,j),bc);
bin1 = bin1/n1;
bin1 = cumsum(bin1);

% Calculation of z- or d-statistic
zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

% determine the sensitivity class (sc)
if zd(j) > czd(3)
elseif zd(j) < czd(5)
```

```
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
    bin1 = cumsum(bin1);

    % Calculation of z- or d-statistic
    zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

    % determine the sensitivity class (sc)
    if zd(j) > czd(3)
    elseif zd(j) < czd(5)
       sc(j) = 3;

    % compute the d-statistic
    bin0 = hist(a0n(:,j),bc);
    bin0 = bin0/n0;
    bin0 = cumsum(bin0);

    bin1 = hist(a1n(:,j),bc);
    bin1 = bin1/n1;
```

```
   bin1 = cumsum(bin1);

   % Calculation of z- or d-statistic
   zd(j) = max((bin0 - bin1).*sign(bin0 - bin1))/n02;

   % determine the sensitivity class (sc)
   if zd(j) > czd(3)
   elseif zd(j) < czd(5)
      sc(j) = 3;
   end

end
    sc_y(:,yr) = sc;               % classification
    zd_y(:,yr) = zd;               % z- or d- statistics
    pr_y(:,yr) = (n1)/(n0 + n1);    % Simple probability
    % Absolute values of parameters for behavior giving simulations
    seg_abso(1:n1,:,yr) = a1n_abso;
    % Absolute values of parameters for non-behavior giving simulations
    seg_abso(n1+1:end,:,yr) = a0n_abso;

end
% Matrix of parameters with sc assigned (the rest will show a zero)
% Matrix of parameters with zd assigned (the rest will show a zero)
% Matrix of parameters with pr assigned (the rest will show a zero)
para_sc = zeros(pg3(end)+len,x);
para_zd = zeros(pg3(end)+len,x);
para_pr = zeros(pg3(end)+len,x);

% Adjust 'to_sample' vector with 'inout' changes
% determine which 'inout' parameters where subject to RSA
to_sampleIO = (pg3(end)+1:pg3(end)+len)';
to_sampleIO = diag(to_sampleIO) * input(end-2,:)';
to_sampleIO(to_sampleIO==0) = [];
% Modified 'to_sample'
to_sampleIO = [to_sample(1:end-x*numIO);to_sampleIO];
%to_sample = [to_sample;to_sampleIO];
% 2.- Parameters values color coded to differenciate B and NB
if modset(20) > 0
    bubbleplot

ano = modset(10);
species = base_ID(modset(17),3);
ParofInt = modset(20);
IndofInt = base_ID(modset(17),2);
BehValue = obje_v;

% Number of simulation that resulted in behavior
Brange = size(a1n,1);
Brange = mean(Brange(Brange~=0));

% Position of the parameter of interest among sampled parameters
sp_par = find(to_sampleIO==ParofInt);

% [seg_abso = zeros(nsim,npar,x)]
Bpar = seg_abso(1:Brange,sp_par,ano);
NBpar = seg_abso(Brange+1:end,sp_par,ano);
% mean(Bpar)
% mean(NBpar)
% Partially rebuilding 'abso2' format [nsim,1]
abso2reb = zeros(size(norm2,2),1);
abso2reb(beh==1) = Bpar;
```

```
abso2reb(beh==0) = NBpar;

% Plot value of behavior [seg_abso = zeros(nsim,npar,x)]
figure('name',strcat('Indicator versus Parameter ',num2str(modset(20))));
scatter(1:Brange, Bpar,'b'); % plot the generation of d
xlabel('Sampling attempt');ylabel('Parameter value');
hold on;
scatter(Brange+1:size(norm2,2),NBpar,'r');%
hold off;

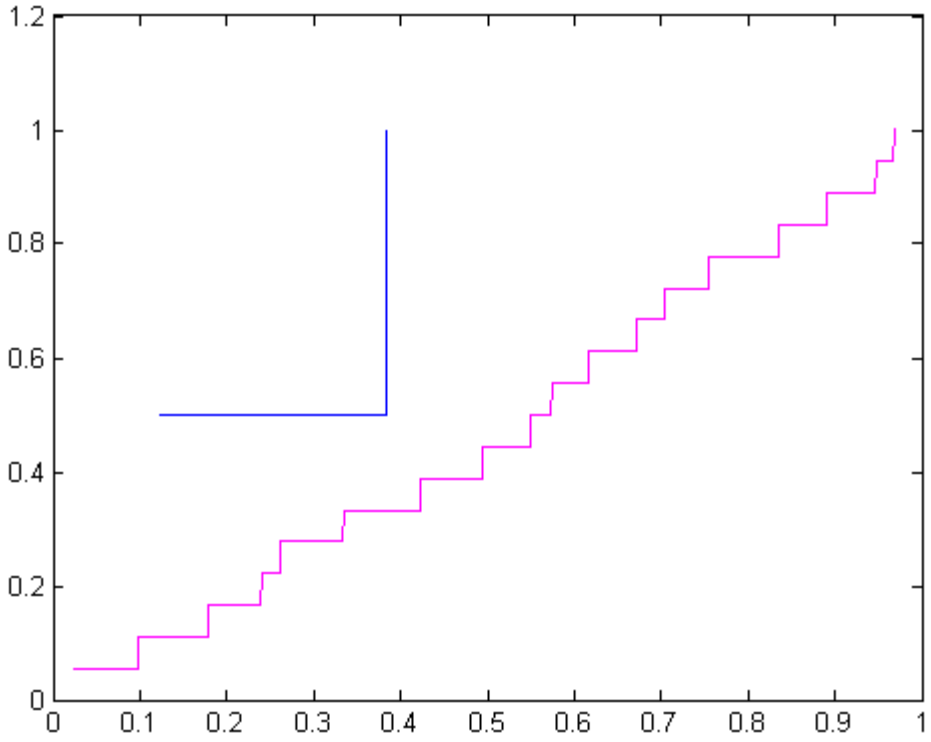
% Plotting parameter value versus indicator value
%indimatRSA  = zeros(x,nmat,nindi,nsim);
perIndi = permute(indimatRSA(ano,species,IndofInt,:),[4 3 2 1]);

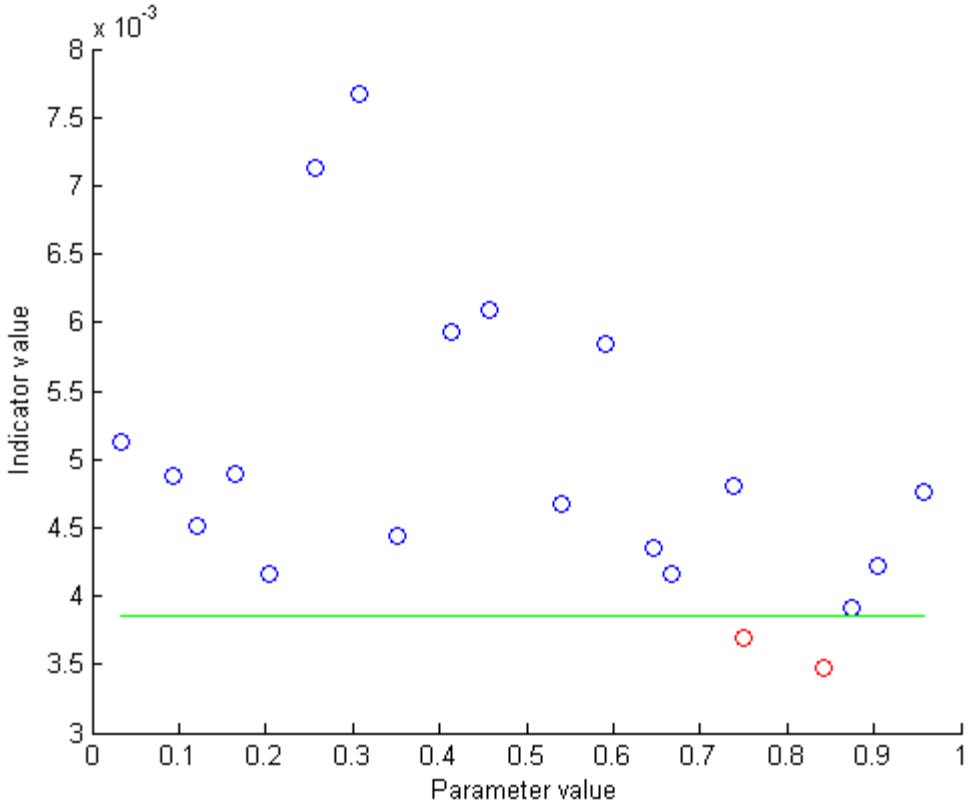
% Plotting B and NB with different colors
scatter(abso2reb(beh==1),perIndi(beh==1),'b')
xlabel('Parameter value');ylabel('Indicator value');
hold on;
scatter(abso2reb(beh==0)',perIndi(beh==0),'r')
hold on
plot(abso2reb,ones(size(abso2reb,1))*BehValue,'g')
hold off;
end
% Classify all parameters
para_sc(to_sampleIO,:) = sc_y;
para_zd(to_sampleIO,:) = zd_y;
para_pr(to_sampleIO,:) = pr_y;

% Rank uncertainty sources
if modset(21) == 1
else
    para_un = 0;
end
% user
AAA = 'msaout'; %strcat(num2str(mat_sce(2:5,grand+1)))'; %'test'; %'rsa_We'; %
save(AAA, 'flowmatRSA', 'indimatRSA','statoutRSA', 'para_zd', 'para_pr', ...
    'para_sc', 'seg_abso', 'to_sampleIO', 'norm2', 'abso2','pg3', 'len',...
    'x', 'nsim', 'beh', 'nflow','infbodRSA','effbodRSA');

save msalatin abso norm to_sample to_fix beh -double

    %************************************************************************
    % CHECKING ERROR LOG DURING RSA
    %************************************************************************
    sumError = [(1:modset(6))' zeros(modset(6),1)];    % 5 major flows x 5[mean, SD]
    sumError(:,2) = sum(errorsRSA,2);
    save('sumError','sumError');
end
% Export some results to the export file (EXCEL)
waitbar(6/10,wh) % waiting bar
if modset(9) == 1
end
% computation here %
waitbar(10/10,wh)
pause(2)
diary off
close(wh)
toc
Elapsed time is 47.571411 seconds.
```





Published with MATLAB® R2013a

root