Guide to data and code

Matlab Data Files

The data are stored in separate folders for each cell. For example, folder rr171019 contains data for this cell (recorded on 19th October 2017).

The file GreenFilter.mat contains a single variable, green_filter_left_eye, which specifies whether the green filter was over the left eye (=1) or right eye (=0).

The other mat files contain data in the specified condition, e.g. rr160818_brightbar_on.mat contains data about the "on" response to bright bars, rr170628_darkbar_off.mat contains data about the "off" response to dark bars.

The data is contained in 4 variables, whose size depends on "nreps", the number of times each stimulus sequence was repeated (generally limited by how long Ronny was able to hold the cell).

The variable "background", size 1 x nreps, is the response when no bars were presented anywhere on the screen. To compute the background, we counted spikes in an 800ms window preceding each stimulus sequence, but the number of spikes was then rescaled by t/800 for comparison with the response to bars, where t is the time-window used for bars (see below). This is why the numbers in variable "background" are not in general integers.

The array *Buffer_blue*, size 6 x nreps, is the response to monocular bars seen by the eye with the GREEN (!!!) filter. It is named *Buffer_blue* because for dark bars, the bar seen by the eye with the green filter is the one which appears blue on the screen, when viewed without glasses by a human observer. Thus, if the green filter is over the left eye, then Buffer_blue gives the response to monocular bars in the left eye.

Similarly the array *Buffer_green*, size 6 x nreps, is the response to monocular seen by the eye with BLUE (!!!) filter.

The array *spikeCount_binoc* has size 6 x 6 x nreps. It gives the response to binocular bars. The first dimension in *spikeCount_binoc* describes the position of the bar stimulus seen by the eye with the green filter (i.e. the same as *Buffer_blue*), while the second dimension in *spikeCount_binoc* is bar stimulus seen by the eye with the blue filter (same as *Buffer_green)*. The third dimension corresponds to the repetition.

Thus in each file, there are data for 49 conditions: 6 responses to monocular bars at one of 6 positions in the left eye, another 6 responses to monocular bars in the right eye, 36 responses to binocular bars, plus the background response.

For "on" responses, the response to a bar is defined as the number of spikes fired in a 250ms window starting immediately at the bar onset.

For "off" responses, the response to a bar is defined as the number of spikes fired in a 200ms window starting immediately at the bar offset.

This applies to the spike counts stored in *spikeCount_binoc*, *Buffer_blue* and *Buffer_green*.

The spike counts were then normalised trial-by-trial to account for drifts in response rate. Data were collected in repetitions where all 48 stimuli were presented. In each repetition, we divided by the

maximum number of spikes seen in any condition. This normalisation is carried out in function FitACell.m where the code is commented to describe it.

Matlab Code

The program you want to run is called FitACell.m. This reads in neuronal data, plots it (both raw and upsampled), and also fits a model and plots that.

FitACell.m begins with two variables which you can change to control what cell is plotted:

```
cellname = 'rr151123'; % identifies cell.
condition = 'darkbar_on'; % specifies condition
```

Change these to plot different cells, or different stimuli/analyses for the same cell.

FitACell.m reads in the Matlab data file for the cell, and writes it into a convenient structure called neuronresponse. This is all fairly well commented in FitACell.m. The function PlotNeuronalData then plots this in the format used in the paper:

```
PlotNeuronalData(neuronresponse)
```

The function FitModel does the fitting. You can call it with just neuronresponse as a sole argument, but the optimisation is non-convex so the initial parameters are rather critical. I therefore did it this way - first fitted the L and R RFs without fitting the output exponent (so 12 free parameters), and set the initial guesses for the RFs to 1 at all bar locations:
model12 = FitModel(neuronresponse,ones(1,12))

Then I used the RFs thus found as the initial guess when fitting a full 13-parameter model including the output exponent:
model13 = FitModel(neuronresponse,[model12.RFL model12.RFR 1])

To plot a model fit, just pass its "response" field to PlotNeuronalData, e.g.
PlotNeuronalData(model13.response)

If you want to see the model's fitted RFs, use
PlotModelRFs(model13)

The details behind the model are described in detail in the document README_ModelFitting.

If you are trying to understand the code in FitModel.m, it may help you to know that this allows for a non-zero threshold. We initially included this as a parameter before realising that it was

mathematically equivalent to a non-zero tonic input. So, we decided to fix the threshold to 0 and only vary the tonic input.

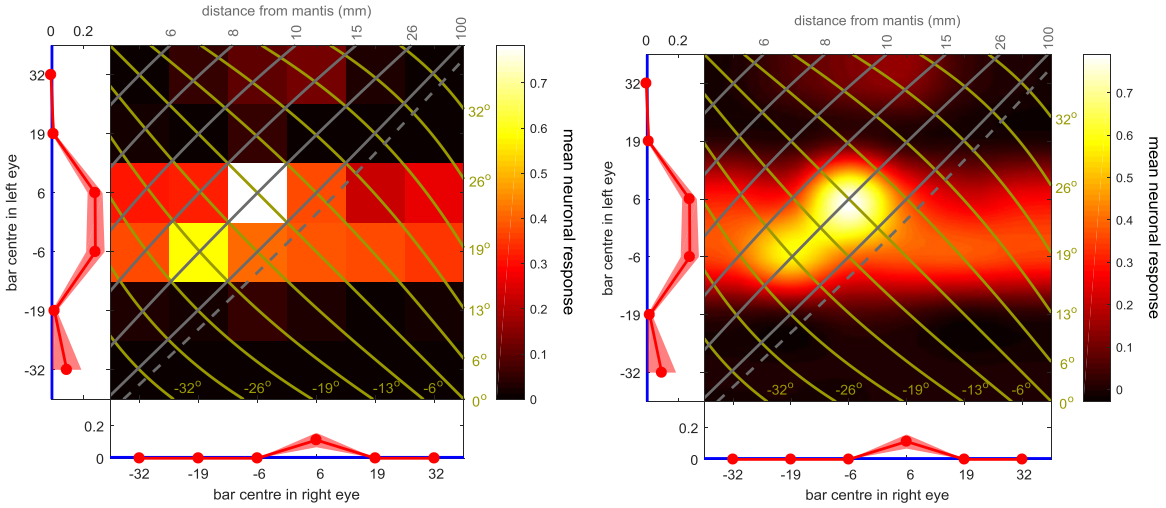
Example output

Here is example output from FitACell.m with

```
cellname = 'rr151123'; % identifies cell.
condition = 'darkbar_on'; % specifies condition
```

Neuronal data

First, we are shown the neuronal data – raw (Figure 1) and upsampled (Figure 2).

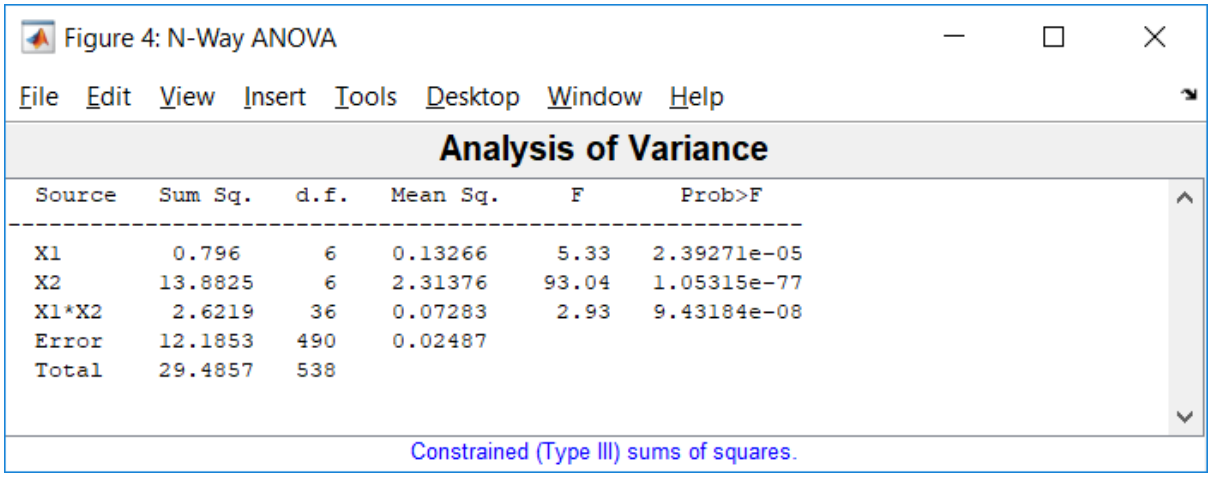


In the command window, we get the output of the ANOVA:

```
Significant main effect of row: p = 0.000024
Significant main effect of column: p = 0.000000
Significant interaction: p = 0.000000
```

So here, both main effects and the interaction are highly significant.

A figure pops up showing the ANOVA output in more detail:



Model fit

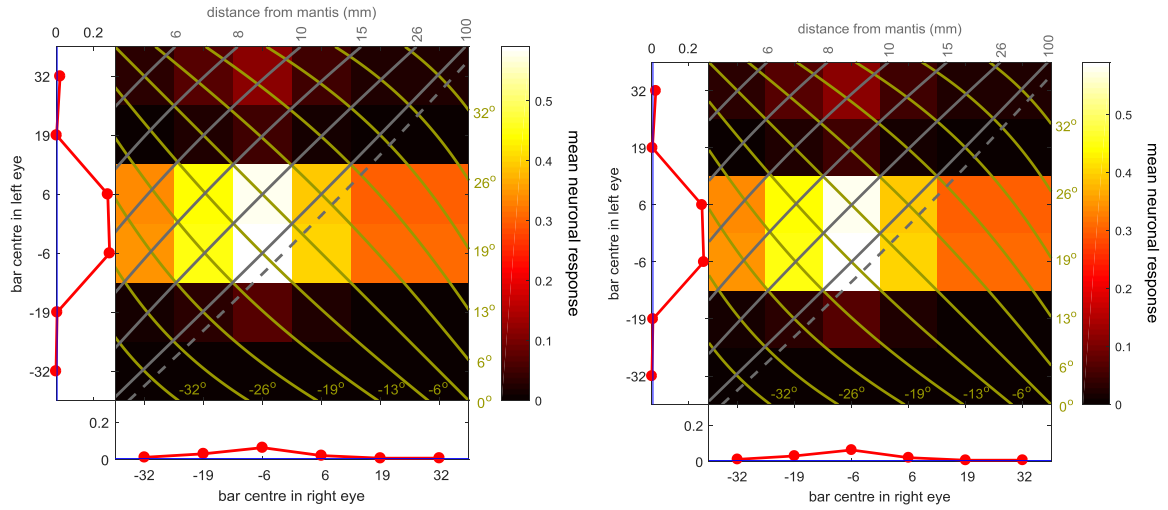
Here the command window shows the results of fitting an initial model with the output exponent fixed to 1:

```
Fitting a model with 12 free parameters, representing monocular responses
only. Tonic input is derived from background and output exponent is assumed
to be 1.
Error of the initial guess provided:  with regularization = 136.321 and
without it = 136.309
Error fitting (this should be lower!): with regularization = 0.216 and
without it = 0.215
Error of initial guess based on monoc responses: with regularization =
0.543 and without it = 0.543
Error after fitting (this should be lower!):    with regularization = 0.208
and without it = 0.207

model12 =
  struct with fields:

                          RFL: [-0.1448 -0.1048 0.3636 0.3361 -0.0549
-6.0038e-04]
                          RFR: [-0.0160 0.0608 0.1405 0.0674 -0.0511
-0.0456]
              outputexponent: 1
                   threshold: 0
                  tonicinput: 0.0057
                    fiterror: 0.2075
                 fiterrorchk: 0.2075
              fiterror_noreg: 0.2072
                    response: [1×1 struct]
    percentVarianceExplained: 86.8370
```

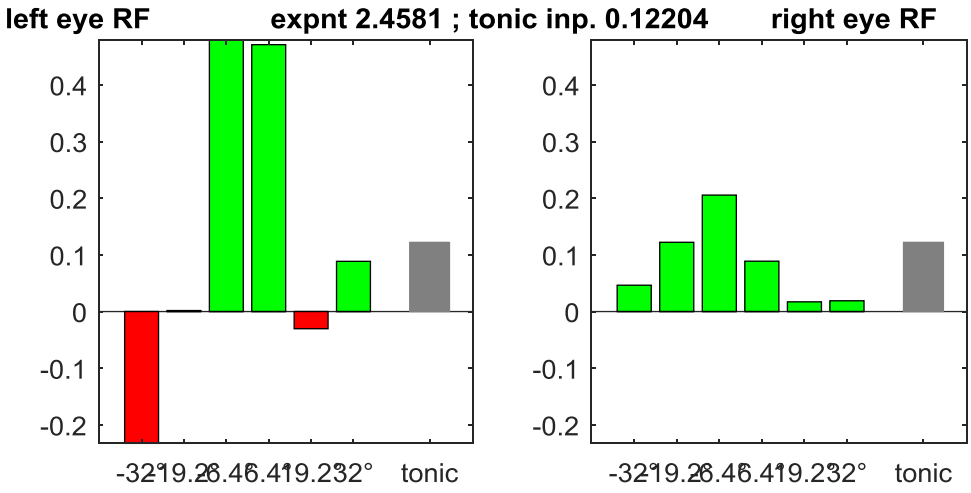
Then we show the results when output exponent is allowed to be non-zero. Note that the description of “13 free parameters” may be a little misleading. We mean the tonic input is fitted to the background rather than to all data at once. The model has 14 parameters.

```
Fitting a model with 13 free parameters, representing monocular
responses and output exponent. Tonic input is derived from
background rather than allowed to be free.
Error of the initial guess provided:  with regularization = 0.209
and without it = 0.207
Error fitting (this should be lower!): with regularization = 0.163
and without it = 0.157
Error of initial guess based on monoc responses: with regularization
= 1.367 and without it = 1.361
Error after fitting (this should be lower!):    with regularization
= 0.167 and without it = 0.160
model13 =
  struct with fields:

                          RFL: [-0.2326 0.0018 0.4796 0.4709 -0.0303
0.0886]
                          RFR: [0.0465 0.1222 0.2055 0.0889 0.0172
0.0191]
              outputexponent: 2.4581
                   threshold: 0
                  tonicinput: 0.1220
                    fiterror: 0.1632
                 fiterrorchk: 0.1632
              fiterror_noreg: 0.1566
                    response: [1×1 struct]
    percentVarianceExplained: 90.0481
```

The program plots the model fits for this final model, again both in raw and upsampled formats:



It also plots the model parameters:



The green and red bars show the parameters L_i , R_i , representing the input from the left, right eye when there is a bar at position i (i=1…6). The grey bar represents *b*, the tonic input which is always present. This can account for a non-zero background response, which is observed in some cells. Its value is also given at the top along with the value of the output exponent.