This code was run using Matlab R2014 and R2016b

From raw videos this is what we did:

Initial Pre-processing

1. Go to Matlab-GCaMP\_image folder and run “processFast()”. From this, we get the Processed\_ROIs\_\*.mat to get the ROI traces.
2. Add a “background ROI” using “Add\_background\_ROIs.m.” Essentially a random ROI.
3. Run “ROIstoVideos.m” to do motion correction to original video, then apply the ROI mask and do trace extraction to those motion corrected videos to get the traces.
4. Run “fullRUpdate.m” on the output of the motion corrected and extracted traces to get the final trace files. This requires the functions “remove\_overlap.m” and “accu\_pca\_filtered.m”

However, when we ran new mice for the review of the paper, we wanted to ensure we had high quality ROIs. To improve our confidence that each ROI we were looking at was actually a cell, we went through and manually selected ROIs by clicking a circle on each region that looked to be a cell.

Pre-processing for Circle ROIs

1. Circle ROIs selected from “roi\_overlay.fig.” This figure was generated from the raw, motion corrected videos that were generated from the above output by taking a maximum projection across the corresponding videos.
2. For ROIs selected as a roi-class, they were converted to a CellList using roi2structure\_conversionScript.m. Most ROIs used and selected as circles did not require the use of this script. The final ROIs used are in the folder of the respective session named “circleROIs.mat”
3. All traces were extracted from using “batchExtractTraces.m” which requires “extract\_trace.m” in the same folder. The output files are named “circleTraces\_\*.mat.”

Figure 1

The only code for Figure 1 was to generate Figure 1C. The script used is called “AverageCal\_All.m.” The raw data used is in the saved figures (1-5).

The rest of the figure was done by inputting and moving around images.

Figure 2

A – No code to generate these figures. Just made from a max projection across the corresponding, motion corrected videos.

B,C – Prior to generating the other plots, the traces need to be binarized. This is done using the “./Figure2/fullRUpdate\_All\_CircleROIs.m.” After generating the binarization codes, they can then be plotted with the Jupyter notebooks (./Figure2/TBIStandardPlots-Current.ipynb).

For Figure 2, the traces used were saved in the “fnormtrace” field in the matlab structure. Individual traces were selected in a semi-random manner by picking at roughly uniformly distributed cell numbers and hand picking from there to find representative cells.

Figure 3

A, B – The spatial maps were generated using the code called “plotSpatialUpDownMap.m.” The line and distribution plots are generated using the script “plotExampleDistributions.m” for the appropriate data.

C – Generated by the script “plotMeanValues.m”

D – Generated by the script “plotMeanValuesPopulation.m.”

Intermediate steps for statistics and additional plotting

The output for “plotMeanValuesPopulation.m” called “sigpopulation\_\*.mat” is necessary for the script compareEleSupIncDec.m.

The output for “activityChanges.m” the output is called “activityResults\_ \*.mat”.

The script “compareEleSupIncDec.m” can be run to perform statistical tests that pull and compare the number of elevated, suppressed, increased, and decreased cells from “sigpopulation\_ \*.mat” and “activityResults\_ \*.mat” together. There are no plots for this script, but it was used to generate the necessary values.

E – These plots are figures 1&2 generated with the script “RearrangeForStats.m.” This script requires “sigpopulation\_\*.mat” to make those plots.

A few FileExchange functions were used for generating these plots.

Patchline was obtained from: <https://www.mathworks.com/matlabcentral/fileexchange/36953-patchline?focused=6794102&tab=function>

Errorbar\_groups was obtained from:

<https://www.mathworks.com/matlabcentral/fileexchange/47250-pierremegevand-errorbar-groups>

Figure 4

A,B – Plots for these were generated using the same Jupyter Notebook for Figure 2 called “TBIStandardPlots-Current.ipynb” but instead of using the “fnormtrace” field in the matlab structure, the “dftrace” field was used in this structure.

C – Both figures were generated from the file “PlotDifferences.m.” Figures 1&2 from this script are what were used in figure 4C. This requires the saved workspace output from “SaveDataStructure.m.”

D – Generated using the same script as in Figure 3E. Figures 3&4 come from “RearrangeForStats.m” and are used in 4D here.

Figure 5

Both figures are generated from the file “PlotDifferences.m” that is used to generate Figure 4C. Figures 9&10 are the corresponding figures for these plots. Note, the usage of the function ‘scatter’ in this script works on Matlab R2016b. It does not work on Matlab 2013a. The function must have changed sometime between those Matlab versions.

Figure 6

This figure was generated by taking microscope images and arranging them in CorelDraw.