**Pipeline for Running S-Probe Analysis**

Neel Agarwal (07/2023)

Sessions:

* c20230531007: this was the session where only half the probe worked
* c20230602007: full session but weaker tuning and visually driven cells than the next 2
* **c20230607004**: this was the really good session run on a black stimulus - use this one for your walkthrough analysis
* c20230607005**:** run directly after the previous session but in white. Still a very good session but it’s questionable whether we can actually call this a separate session from the previous

Random Notes:

* When looking at units: unit 1 is the unsorted spikes, unit
* to force quit when Matlab is running hit command + .
* when recording an S-Probe data set: lower the probe with the thresholds set to -10mV then once you get a good signal on each contact switch to a threshold of -15mV (so that when we run analysis on unsorted spikes, we can eliminate more of the noise)
* when sorting on Plexon Offline Sorter, sort all identifiable single units first. Then reorganizing which of the units you are going to place in the first position (on Plexon - the yellow spike) by weighing the number of the spikes you have, the ISI, the best waveform shape, the most stable waveform across the session (to negate drift).

**WHILE LOWERING THE PROBE (DURING RECORDING)**

Notes:

* Could be useful to first put the dual drive on the chamber because we are more familiar with how those drive coordinates map onto the RF map. If you do this make a slight mark by pressing the guide tube down onto the dura then quickly switch the single drive and line it up to match that guide tube mark.
* Generally when lowering the probe we go in increments of 500microns while someone (Yasmine) is watching the chamber looking for CSF/buckling of the probe.
* When watching the spike panel as the probe is lowering, we would expect spikes to appear first on channel 32 then following this order - **[32, 30, 28, 26, 24, 22, 20, 18, 16, 14, 12, 10, 8, 6, 4, 2, 31, 29, 27, 25, 23, 21, 19, 17, 15, 13, 11, 9, 7, 5, 3, 1]**
* When running finder to get the general idea of the RF of that particular column. You should sample channels 28 (bottom), 2 (middle), 5 (top) on the spike panel.
* When running the mapper experiment, it could be useful to put tape on the monitor that displays the stimulus in places you hear the hash get driven by presentation
* **YOU NEED TO WAIT 30 MINUTES FROM THE TIME THE PROBE IS IN ITS FINAL DEPTH AND THE TIME YOU BEGIN YOUR SHAPES AND BAR EXPERIMENT**

circleflashAnalysis/**parserCircleFlash32.m :** use this protocol for a circle flash file that you record after lowering the probe and getting a good signal on every contact. This protocol can help you determine whether the unsorted units prefer a white to black stimulus (albeit weakly) and will calculate the LFP and CSD across the probe. This can controversially be used down the road to tell where layer 4 lives on the probe. ALSO USEFUL TO RERUN THIS PROTOCOL AT THE END OF THE SESSION SO THAT YOU CAN COMPARE HOW MUCH THE PROBE MOVED ACROSS THE LENGTH OF YOUR EXPERIMENT.

**POST HOC**

1. Sort the continuous data into spike timestamps on Plexon Offline Sorter. Filter using 500Hz Butterworth filter and detect waveforms using a -2.7 sigma threshold (I.e., 2.7 standard deviations above the noise).

2. Save the file on Plexon Offline Sorter in the .nex format. Ensure to export only the spike timestamps that came from continuous data filtering/detecting. Make sure that you also export the unsorted spikes and make sure you check the box that actual sends the waveform shapes. Save the nex filename as [filename, \_sorted.nex] ex. c20230607005\_sorted.nex

3. Use the **parser\_ShapesBars32.m** Matlab script to take the .nex, .nev and .ns4 of your session and turn it into a single .mat that includes the parsed oData and pData.

(Note moving forward for subsequent analyses you will be loading in the .mat file)

YOU CAN DECIDE TO MODEL FIT NOW OR LATER - how to do it is explained under this indent

**CURVATURE MODEL FIT**

a. Use **fitCModel.m** script to fit the curvature model on your shape data. It requires that you first create a folder called [filename\_modelFit] on engram in the data section of Blackrock (ex. c20230607005\_modelFit) where the output will live as well. Currently the fit is set to 10 iterations with no cross validation but can be changed depending on the stage of analysis you are at (refer to Yasmine to do this). This process of fitting takes a lot of time so you should run this overnight and on one of the rig MacBooks (computer with the most RAM in the lab). You can also choose to batch your analysis -> go to line 46 and change the indexing on unitIdentities(n:m).

b. After completely finishing the curvature model fit for all the cells in the session. Use **mergeCModelFit.m** to concatenate all of your currently separate fits (for each neuron) into a single .at file which will be named filename\_cModelFit.mat on engram (ex. c20230607005\_cModelFit.mat). Now all the curvature model fit parameters and information will be accesible by loading this .mat file.

c. Use **checkCModelFit.m** to review each of your curvature model fits. The script will plot a single units real data and the model fit over it (as well as some curves about the cells curvatures response preference). Use it to double check whether the model fits the data well and make notes of certain patterns/etc.. The channel and the unit of the cell (i.e., the identity of the unit) will be placed as the figure title in the format channel\_unit.

**VM MODEL FIT**

d. Use **fitVmModel.m** script to fit the von Mises function on your bar data. This fitting is not so computationally intensive so can be run on just about any computer. This script combines both processes for the curvature model fit so the output will be a .mat file called [filename\_vmModelFit.mat] (ex. c20230607005\_vmModelFit.mat). Now all the vm model fit parameters and information will be accesible by loading this .mat file.

e. Use **checkVmModelFit.m** to review each of your von Mises model fits. The script will plot a single units real data and the model fit over it . Use it to double check whether the model fits the data well and make notes of certain patterns/etc.. The channel and the unit of the cell (i.e., the identity of the unit) will be placed as the figure title in the format channel\_unit.

4. Use the **tuningCluster.m** Matlab script to create 8 plots that plots the tuning of all the units from all 32 channels. Messy plot because several units may be plotted on top of each other (depending if there are multiple isolated neurons on a contact). FIRST PASS -> keep the plots opened or saved so that you can use them as a guide for the next step.

5. Using the **plotSingleUnitTuningPsthWaveform.m** script, a single unit on a single channel - to visualize tuning, rasters/psths, latency and waveform. You can pick out the best unit on each contact with this method (we did that in early analysis but have decided that it’s more meaningful to look at all units for population analyses -> clusters)

**Specific Analyses Scripts**

**Notes for Yasmine**

- I’ve noticed that for some looping of all my neurons, I use the unitIdentites and sometimes I loop first through 32 channels then through the units. I do the second when I want to retain the channel and unit structure for whatever analysis I’m doing. For example, if I want a 2D plot that has channel on the Y axis, latency on the x, and multiple point per y value that represent units.

barAnalysis

- **plotOrientationTuningPolarAcrossProbe.m** : plots simple tuning curves for bar orientation but in polar coordinates. Each subplot represents a single channel (titled) and there are multiple units on each subplot each represented with a different color.

- **plotBarSelectivityAcrossProbe.m** : using the vmModelFit, plots the orientation selectivity across the length of the probe while taking bandwidth in account. You need to have selected the set of cells where the model fit actually fits the real data prior (use **checkVmModelFit.m**).

- **plotOrientationSelectivityPolarAcrossProbe.m** : using vmModelFit, plots the orientation selectivity across the probe in polar coordinates (doesn’t look at bandwidth). This is a population level plot. The data is doubled to span 0-360 degrees by duplicating each orientation selectivity value (0-157.5) and adding 180. The center of the plot is the top of the probe and the edge is the bottom of the probe.

- **plotPairWiseCorrBarTuningMatrix.m** : (PROBLEMATIC ANALYSIS) plots a heat map of the correlation in bar tuning across all pairwise combinations of units in the session. This analysis isn’t the best because orientation is circular and not linear. i.e., the correlation doesn’t take into account if the last values of the firing rate list for unit 1 is similar to the first value of the firing rate list for unit 2.

curvatureAnalysis

\*\*\*\* Function \*\*\*\*

- **plotModelFitTuning.m :** function that is called on in some of the scripts writing within the shapes and bars analysis pipeline. It’s what to use when you want to plot the real shape data with the curvature model fit over it. It also plots features of the stimulus the cell best responds to (angular position, primary curvature, cw curvature, cow curvature). You can refer to **checkCModelFit.m** to see how it’s used.

\*\*\*\* Scripts \*\*\*\*

**- plotPrimaryCurvatureXAngularPositionAcrossProbe.m : VERY INSIGHTFUL ANALYSIS (YASMINE THIS IS THE 3D ELLIPSE PLOT) -** plots the preferred primary curvature and angular position with their respective sigmas as transparent ellipses. The 3D version is easier to understand. Currently is written to plot the E neurons in blue, the I neurons in red, the unclassified (qualitatively done to be trimodal or peak before trough neurone) -> this can easily be changed to make them all transparent grey if you want. I like looking at the view were you only see the x and y axis so all the transparent ellipses are stacked on top of each other (to look for clustering or laminar consistencies).

- **plotShapeAngularPositionXBarOrientation.m :** plots the orientation selectivity based on the vmModelFit on the x axis and preferred angular position for shapes on the y for a particular neuron. NOTE: ANGULAR POSITION 0 MAY NOT BE EQUIVALENT TO BAR ORIENTATION 0 - MAKES THIS ANALYSIS UNFINISHED/WEAK

**- plot2DcorrX4Dcorr.m :** plots the 2D corr value on x and 4D correlation value on y for a particular neuron (these values represent how well our 2D and 4D model fit the data). Used to determine which model fit the data better and generally what type of correlation values we are getting from the model for this particular session. There is a vertical line that represents the median 2D model fit correlation value and horizontal line for 4D median.

- **plotPrimaryXCwXCcwCurvaturesAcrossProbe.m :** Used to plot the preferred primary, clockwise and counterclockwise features of the neuron in 3D based on the curvature model fit. The points are different colours which spans the Matlab hot colormap to represent which specific units channel. This analysis script hasn’t been used much by me yet but it seems like it could be a useful tool in the future.

- **plotShapeAngularPositionAndBarOrientationAcrossProbe.m** : plots the orientation selectivity for bars with bandwidth as error bars across the probe on the first subplot (based on vm model fit).The second subplot is the angular position preference for shapes with its sigma as error bars across the probe (based on curvature model fit). NOTE: ANGULAR POSITION 0 MAY NOT BE EQUIVALENT TO BAR ORIENTATION 0 - MAKES THIS ANALYSIS UNFINISHED/WEAK

- **plotPolarRepresentationCurvatureModel.m :** uses polar plots to visualize the 4 main parameters that come out of the 4D curvature model fit (angular position, primary, clockwise, counterclockwise curvature). This is an abstract representation created to try and visualize the 4 dimensions of the model fit in a meaningful way. There are 32 polar plots representing the channels and each subplot has all the units that were isolated on that channel. To interpret - look for the center of the 3 points - on the radius axis that’s primary curvature and on the degree axis that’s angular position. Then the points that surround it represent the cw and ccw curvatures (placed 45 degrees to cw or cow to the angular position). There is a circle plotted at the center of the plot at radius = 0 which represents the boundary between convex curvature features and concave curvature features. (-1 concave, +1 convex).

- **plotSingleUnitCModelFitTuning.m :** plots the real shape tuning data and the predicted curvature model fit data of a particular single unit on a specific channel. Essentially calls the function **plotModelFitTuning.m** explained above but here you only have to input the channel and unit identity of your neuron of interest.

waveformAnalysis

NOTE: to do waveform analyses you need to have looked through all waveforms and eliminated waveforms that are not bimodal and don’t have a through before a peak. Once you have qualitatively done this, write the unit identities into the **getEICheck.m** function in the functionsNA folder - follow the nomenclature for the other filenames that this was done for.

NOTE: we call a waveform I if spike width is less than or equal to 0.2ms and E if greater

NOTE: noticeably more inhibitory neurons than excitatory neurons

**saveEvIcModelFit.m :** saves the curvature model fit plots but separates units depending on whether they are E v I based on spike width and qualitative removal of spike waveforms that aren’t bimodal and through-peak organization. Useful for determining the differences in putative E and I tuning, model parameters, feature preferences, whether one has more broad tuning, etc.

**plotSpikeWaveformsAcrossProbe.m :** plots the waveforms that were determined usable for waveform analysis across the probe. Organization 4 x 8 subplots with the channel number represented as the title. The waveforms are also color coated based on whether they are E or I.

**plotSpikeWaveformsPopulation.m :** plots all the waveforms of every unit in the session on top of each other. Inhibitory neurons are red, excitatory are blue and other are grey. The total number that belongs to each category is put as the title of the figure.

**plotSpikeWidthDistribution.m :** calculates the spike width of all the isolated units within the session then uses a 0.01 ms bin width to plot the spike width distribution (histogram). Used to see the population level spike widths - we would expect the distribution to be bimodal that separates at 0.2ms which is the widely accepted boundary between E v I neurons.

plotSpikeWaveformsAcrossProbe.m and plotSpikeWaveformsPopulation.m in the waveformAnalysis folder also include the ones that don’t meet the EvI criteria. The others are in grey (that don’t fall into bimodal and through then peak)

Also - there’s a script called checkEachSingleUnit.m in the general sProbeNA folder that cycles through each individual single unit and plots tuning, psth and waveform so you can check all the data.

­