Assignment #3
Processing of visually evoked neuronal activity Part 2: Spike sorting

Intro for HW3

Extracellular potentials in the brain are due to weighted sums of contributions from transmembrane currents following neural activity. The extracellular neural recordings contain periodic waveforms generated by the firing neurons around the electrode recording site. The magnitude and shape of the extracellular spike vary with cell type and distance to recording site. Those spikes can be detected and classified to determine the firing rates of different neurons. As we covered in Assignment #2 introduction, many recording conditions also result in high noise levels that increase the difficulty of distinguishing the separate neuronal waveforms. Recorded neural spikes are superimposed with neuron noise from distant neurons, electrode-electrolyte interface noise, tissue thermal noise, and electronic noise. Most of the noise show frequency dependency and noise models are used to develop noise removal techniques to remove some frequencies or frequency bands.

Even after the data has been filtered there is a certain amount of noise that interferes with the detection of action potential waveforms. The neurons that are close enough or can produce a large enough voltage can stand above a given threshold for the noise distribution. The threshold crossings of those action potentials can be detected. The time at which the threshold crossing of action potential occurred can be used to quantify firing rates. We can also define an interval around the threshold crossing and look at the “snippets” to appreciate the action potential shape. The action potentials can have different waveforms depending on the subtype of neuron, and the cell’s spatial orientation with respect to the electrode. Careful consideration of these factors can help identify different neurons and their respective firing patterns.

The goal of this assignment is to use filtering techniques covered in the last assignment and identify the time points of action potentials detectable above the noise threshold. Next, using a defined time interval around those threshold crossings, you should be able to extract and plot the action potential waveforms. The processing steps for identifying the time points and a plotting window around the maximum deflection of each spike are outlined below.

Load the assigned dataset:

Load 0900702VisuallyEvoked.mat
(or whichever dataset was assigned to you in Assignment #2)

The sample rate is the variable ‘samprate’ in Hz
TrigON is the time when a visual stimulus was presented
TrigOFF is when the visual stimulus was turned off
Wideband_data contains the raw neural recording (the preamp headstage had a highpass 2Hz filter)

- Cells are organized from top to bottom (ie, Cell 1 is at the surface of the cortex and Cell 16 is the deepest in CA1). Each cell should have a dimension of 1 x # where # is the total number of data points collected. The data is stored as Volts.

C. Spike plots

1. Find the mean, standard deviation of each channel

```
help mean
help std
```

2. Plot a histogram of the data for each channel and report the channel #, mean, and std (label units) in each subplot title. Save/print figure to turn in. Try plotting the Y-axis in log.

3. Pick a detection threshold (two tail, 1 direction) in standard deviations (eg 2x STD) for 99.90-99.994%. Determine the negative threshold for each channel [mean-(your detection threshold value in STDs)*std)]

4. For each channel, we need to create an index and find the time point when this threshold was crossed

```
help find
```

5. Now, we need to go through each element in the index

```
help for
help numel
```

6. Given what we know about voltage-gated ion channels (max firing rate is 1,000Hz), and nyquis frequency, sampling rate, determine the size of a snippet window for capturing action potentials. You may want to use a variable here, so that you can easily adjust it later on. We need to ignore threshold crossings that occur right at the beginning or the very end of our recording session, because we won't have enough information to fill our window.

- Eg. If our window is 11 samples wide and a threshold crossing occurs at sample 3. If we try to get 5 samples after sample 3 and 5 samples before sample 3, the script would error, because we are missing 3 data points before the recording started.
- You can do this using if / elseif or use find again or another other ways that may or may not require more lines of coding.

7. As in the example, we now need to create a short array, the size of our snippet window around our point of interest. We need to use the index from above and extract the data from the spike filtered data

'your spike flitered data variable'(index-'window component before':index+'window component after');

Note, you can make your window asymmetrical.

8. VG-Na kinetics (negative deflection) are much more uniform than K+ channels (positive deflection), so it's best to align all exctracellular spikes at their valley. Use the function 'min' to extract the 'minimum value', and 'minimum position/index'.

```
help min
```

9. Now we need to re-adjust our window around the 'minimum position' as in step 8.

'your spike flitered data variable'('minimum position'-window component before: 'minimum position'+window component after);

10. We also want to convert the 'minimum position' to a time stamp.

To do this, you will need to start at your index, adjust for the 'window component before' then move forward to the 'minimum position'.

Grab this value from your 'filtered spike data' using filtered spike data'(new position) and check to see if this voltage is identical to the 'minimum value' you extracted in step #9. If it is not, you might be missing a '+1' or '-1' somewhere depending on how you counted/set your window. You may also want to convert from samples to time for your timestamp ('sample position'/samprate)

11. Check to see if this timestamp is identical to the timestamp generated by your previous index. (in order to do this, you might need to initialize a timestamp checking variable before your for loop (from step 6#). Initialize it with a value that you know the first timestamp will never be (like 0).

12. If your timestamps are different: a) enter timestamp into a timestamp matrix for that channel. b) enter the corresponding snippet into a snippet matrix for that channel.

You might consider using a counter variable (again initiate before the for loop) that increases by 1 every time you find a unique timestamp. You can then use the counter variable to identify the row for appending the timestamp or snippet into.

'Matrixvariable'('counter variable', :)

13. Optional: include positive deflection threshold, aligned using valley, and checked to insure only unique timestamps are stored (from negative deflection)

14. Plot the pile plots of all the snippets for each channel.

Extra Credit:

D. Noise Floor

Fast firing activity from high amplitude spikes can skew your standard deviation calculation. For SNR and noisefloor quantification, we want the "standard deviation" to characterize the Gaussian noise. In order to do this, we need to remove the snippets from our spike waveforms from our raw data and recalculate the standard deviation.

1) We need to know which columns in your 'filtered spike data' correspond to extracted snippets

You can:

a) Create a column index in step C.13
b) Unpack the timestamps to identify the relevant column #s.

2) Organize the information into a single row (use 'cat' if you need to)

- Eg. If your window is 5 samples long and spikes occurred centered around column 10 18 and 36, your array should look something like:
  8 9 10 11 12 16 17 18 19 20 34 35 36 37 38

Note, all values in the array should be non-zero positive integers

3) Find the last column in your 'filtered spike data' and create a linear 'temp index'

```
help numel
help size
=1:#;
```

4) Now use setdiff to retrieve column numbers that were not used as part of spike snippets

```
help setdiff
```

5) collect the non-spike data by using the output from #4 to retrieve the corresponding voltage values from the ‘filtered spike data’

6) Find the mean, standard deviation of each channel again.

```
help mean
help std
```

7) Plot another histogram of the data for each channel and report the channel #, mean, and std (label units) in each subplot title. Note if the histogram appears more Gaussian or not. Save/print figure to turn in.

Note, you may find that you need to readjust the snippet window duration to optimize the results in C and D. If your window is too large, you might capture multiple spikes inside your window, or if your window is too short, you might see that some of your spike is still calculated as part of your noise floor.

8) Organize each channel into structure and/or cells. Plot a 3-10 second window of each channel and report the channel #, detection threshold in microvolts, and the voltage of the noisefloor (2std). Plot the threshold and comment on if the threshold is too high or too low or just about right.