**Lab 5: Interpretation of neural recordings**

# Introduction

The first four labs introduced computational approaches to studying electrophysiology. In this lab, we will begin to use and examine tools that are used to interpret experimentally acquired data: a model for characterizing tissue response, rudimentary approaches for analyzing local field potentials, and a tool for dimensionality reduction.

# Software

This lab must be completed using MATLAB.

# Part 1) Estimating effects of tissue response from impedance spectra

The insertion and presence of conventional recording electrodes in nervous tissue causes physical damage that leads to tissue responses, such as gliosis and edema. The resulting encapsulation of the electrode is known to significantly affect recording properties. Although it can be impractical to directly observe the tissue response during recording, the electrode boundary model can be used to estimate a number of parameters using readily obtainable data.

In Figure 3D of ‘Complex impedance spectroscopy for monitoring tissue responses to inserted neural implants’, by Justin C Williams, Joseph A Hippensteel, John Dilgen,William Shain, and Daryl R Kipke (2007), there are two impedance spectra of an electrode—one taken immediately after implantation and one seven days later. Using information from the graph, determine the following:

1. Estimate the values of for both lines by extrapolating the first few points at the top of each plot. If , treat it as 0.
2. Estimate for each plot using the same few points at the top of each plot.
3. Calculate for both lines. Use frequency values associated with the topmost point of each plot. (You can figure out the frequency from information in the paper.) Note that (a complex value) can be read directly from the plots.
4. Using the parameters you estimated earlier, calculate the magnitude of the tissue related response and constant phase element at 1 kHz (that is, calculate for ) for both plots. Remember that .
5. What conclusions can you make about the electrode performance and the tissue response at the time of implantation and seven days later?

# Part 2) Feature extraction from ECoG

Interpretation of field potentials is not straightforward. Analysis of even one channel from an ECoG array requires several layers of processing before arriving at meaningful features that can be used for decoding or interpretation. Here, we will examine one channel of data from a 128-channel ECoG array.

1. Load ecogdatasnippet.mat into MATLAB. The data is sampled at 1000 Hz.
2. Apply common average referencing to channel 29. Only use channels that show up in refChannels.
3. For channel 29, extract the three frequency bands analyzed in Pistohl et al (2011), ‘Decoding natural grasp types from human ECoG’. MATLAB has a number of filter design tools that can all be used for this task. Use your favorite bandpass filter.
4. Calculate the power of these signals over time by squaring the voltage at each time point. Use MATLAB’s smooth() function to smooth the features over a 100 ms window and plot over time. This will make task-related activity easier to see.

# Part 3) Dimensionality reduction of spike recordings

Spike waveforms can be treated as high-dimensional data, with each sample along the voltage trace treated as a separate variable describing the waveform. (For example, a 3 ms snippet recorded at 10 kHz would have 30 samples and can be treated as 30-dimensional data.) However, many of the sampled points are often strongly correlated with each other and it is possible to significantly reduce the dimensionality of the data while losing very little information. This can be useful for visualization, spike sorting, and data compression.

One of the simplest and most widely utilized approaches to dimensionality reduction is principal components analysis (PCA). PCA identifies features of the data that capture the greatest variance, which can then be used to approximate the data, such that

Where is a normalized data point, is an eigenvector of data matrix , is the weight of eigenvector for sample , and is the number of eigenvectors used to reconstruct the data. You can think of as a vector that captures a feature in the data, and as the strength of that feature in a given data point . We call the jth *principal component* of data point ­, and the jth *principal eigenvector*. Note that for each eigenvector , there is a corresponding eigenvalue . Oftentimes, it is possible to very accurately reconstruct the data with , where the is the original dimensionality of the data.

1. Load spikes.mat into MATLAB. The file contains 41568 snippets of spikes, each 32 samples long.
2. Normalize the spike traces over the data set such that every point along the trace has a mean of 0 and standard deviation of 1. We do this so that each point along the trace has equal importance in the PCA calculation.
3. Use the MATLAB built-in pca() to calculate the data’s principal components. Use MATLAB’s help function and documentation of pca() to determine the which outputs correspond to , , and . You will need all three in the following steps.

Note: the MATLAB outputs will be in matrix form, while the equation above is written in vector form for ease of interpretation. (That is, MATLAB will provide , , and .)

1. Each eigenvalue is proportional to the amount of variance captured by eigenvector . Determine the number of principal components, , necessary to capture 90% of the variance in the data. (Hint: the cumsum() function might be of help here)
2. Pick a representative spike and plot the top eigenvectors of the data (the eigenvectors corresponding to the largest eigenvalues). Comment on which features are captured by each eigenvector.

Now that you’ve deconstructed the data into principal components, let’s try to reconstruct the original spikes with lower-dimensional data.

1. Using the equation introduced earlier, approximate spike #5, with . That is, reconstruct the spike using every principal component. Un-normalize the reconstructed spike so that it looks like the original waveform. Compare this to the original waveform.
2. Now approximate spike #5 with six principal components. Then four. Plot your results overlaid with the original waveform and comment on the resulting waveforms. Calculate the mean squared error of the approximations.

PCA can also be a useful tool for data clustering and visualization. Since a small number of principal components can capture a large portion of the data’s variance, it is possible to represent differences in the data in a low-dimensional space.

1. Plot the first principal component (the one associated with the largest eigenvalue) of each data point against the second principal component of each. What do you see?

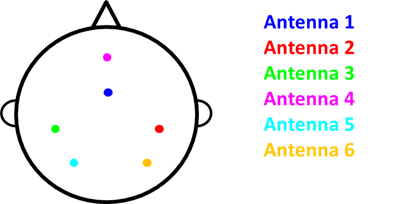
In the next lab, we will use clustering algorithms to programmatically sort this data.

# Part 4) Independent component analysis

Another way to perform data reduction is to run independent component analysis (ICA). ICA finds a linear combination of the data channels for each resulting component. The output of ICA is a weight matrix mapping the channels to components that it found. The ICA method you will be using here tries to minimize mutual information among the resulting components. Mutual information is similar to correlation, but it is between probability distributions instead of the actual signal values. However, mutual information measures independence, which correlation does not. By minimizing mutual information between components, the resulting components from ICA are maximally independent from each other.

We will look at a basic example of running ICA on data. The data was collected using electroencephalography (EEG) electrodes on the scalp of a fake head. The fake (or phantom) head has multiple antennae inside it, which have been set at different frequencies. Unfortunately, recording EEG data mixes the antenna signals together. Your job is to figure out which signal frequency belongs to each antenna. The EEG data (‘eegPhantomDataSnippet.mat’) has 128 electrodes already referenced to the average and sampled at 256 Hz.

1. Open up ‘Lab5\_part4\_ICA.m’ and navigate to its containing folder on your computer. Run ICA using the code provided. This will take ~2 minutes to run. In our setup, we run PCA on the data prior to running ICA, and only the first 60 components will be retained. ICA returns the same number of components as input channels (ex. if we input 128 channels, ICA will try to find 128 independent sources). By using PCA, we are asking for ICA to find only 60 sources. This speeds up the algorithm, and we only care about the first few components.
2. Once ICA finishes, analyze the results of the first 6 components. We analyze ICA by looking at the channel weights and power spectrum for each component. To look at the channel weights, run the cell with the topoplot function. This loads in the location for each EEG electrode and plots the weights for all channels for each component. The resulting plot looks at a head from above, with the triangle at the top of the circle being the nose. Smoother topoplots, with a one positive (red) and one negative (blue) area indicate better ICA results. You should be able to estimate the antenna corresponding to each independent component by looking for the maximum topoplot magnitude (positive or negative).
3. Fill in the missing code to run a Fast Fourier Transform (FFT). Plot the result from 0-60 Hz. What would you expect to see if ICA did a good job separating the original antennae signals? Does what you see match that expectation? Based on your FFT spectra and topoplots, what might be going on?
4. Load in the weight matrix (‘icaweights\_amica’ and ‘icasphere\_amica.mat’) and perform the same analysis methods in steps 2 and 3. This weight matrix was run on this data using a much more sophisticated ICA algorithm that took hours to run on Flux, UM’s supercomputer. Are you now able to tell which antenna generated each signal? Interpret the differences in the topoplots between this and step 2. (Note: the antennae inside the phantom head are dipoles.)
5. Match the antennae numbers below to each peak frequency you found in your FFT results. Show the topoplots and FFT spectra of the first 6 components for whichever method worked better for you. (Hint: The highest antenna frequency is close to 50 Hz.).



# Guidelines for Lab Report (on Labs 5 and 6 together)

*Introduction:* The introduction should be one paragraph long summarizing the motivation for developing the tools used in this lab and what they can be used for, along with a brief summary of everything you will show in this lab report.

*Methods:* From Lab 5, there should be methods paragraphs (and diagrams where necessary) on:

1. Assumptions of the models used and how the models were designed
2. Any signal conditioning methods used prior to analysis
3. The software tools used to implement signals analysis

Include the code as an Appendix to your report. Cite sources for any values used in your models.

*Results:* You should include the following in your Results:

1. The outcome of electrode interface calculations
2. Plot of the features (3 power bands) extracted from the ECoG signal
3. The accuracy and limits of dimensionality reduction by PCA (show plot of 9 PC’s, plot of reconstruction with different numbers of PC’s, and plot of PC1 vs. PC2)
4. Show one figure for the topoplot and one for the FFT spectra, with 6 subplots each

Include all figures produced by MATLAB that could help explain and illustrate your findings.

*Discussion:* Should be 2-3 paragraphs long describing what you could use these models for in the future.

This report will be combined with Lab 6, to create one cohesive report. The report (not including Appendix) should be no longer than 4 pages. Use 12 pt. font and 1.15-1.5 line spacing. If your text is over the 4-page limit with figures, you can move your figures to an appendix section that goes beyond the 4-page limit. However, any text that goes beyond this limit will not be graded, except for figures, figure titles (no captions), and your code.

Please upload your report to Canvas and leave a hard-copy with your GSI in lab. The hard-copy will be graded, so be sure different lines on your plots are distinguishable (using color or different line styles).