Lega Lab: Introduction to Behavioral and EEG Analyses

Contents

Introduction

In this tutorial, you will use the files saved in 'LegaLabTutorial' to perform several common analyses used in our lab. This tutorial is structured to be executed in MATLAB. While this tutorial assumes a basic knowledge of MATLAB or other programming language, even an inexperienced programmer should be able to complete the exercises below with little guidance. A Google search for generic questions such as "How to average values in MATLAB?", "How to loop through files in a folder using MATLAB?", or "How to plot a line in MATLAB?" should return useful suggestions on how to perform these functions. Although this tutorial may initially be challenging if you have not worked with MATLAB before, the projects are organized in increasing complexity. You should be able to build on the skills used in each of the exercises to proceed to the more complex analyses.

If you notice any issues with this tutorial (unclear instructions, typos, etc.), please email jimmygermi@gmail.com. This tutorial is meant to be updated as much as necessary to develop a guide that will help new members of the lab develop the skills necessary to perform complex neural signal analyses. If you have a question, it is likely that someone in the future will too and it would be helpful to answer as many questions as possible in the context of this guide.

Objectives

The primary objective of this tutorial is to familiarize new lab members with the format of our data and the functions in the eeg_toolbox that we use for basic signal processing. By working through each of these exercises, you will be able to develop the skills necessary to perform more sophisticated analyses using MATLAB. The most effective members of our team will be able to independently analyze the data. A key objective of this tutorial is for you to learn how to troubleshoot issues with your code using online resources. These exercises will be most valuable if you try to resolve any issues on your own before reaching out to another member of the lab for help. If you are unable to find solutions to your questions using online resources or this tutorial, you are encouraged to reach out to another member of the lab for guidance.

LegaLabTutorial Folder Contents

In this section, we will go through the contents of the LegaLabTutorial folder that contains the necessary data to complete this tutorial.

LegaLabTutorial

1. **sampleData**
   This folder contains the a subset of our data. The folders within 'sampleData' correspond to individual subjects. For example, folder UT014 contains the data for a subject with the unique subject code 'UT014'. The 'UT' in the subject code indicates that the subject was enrolled in the study at the University of Texas, Southwestern. The subject number is 014. The UT subject codes follows the format 'UT###' where the number indicates the number relative to the first subject enrolled in any of our studies at UT Southwestern. The first subject is UT001, the second subject is UT002 and so forth. Dr. Bradley Lega also collected data during his fellowship at the Cleveland Clinic. The subject code for these subjects follows the format 'CC###'. For the purpose of this tutorial, you were only given data for 'UT###' subjects.

   In order to explain the format of our data, we will go through one of the subject folders.

I **UT014**

This folder contains data for subject UT014.

i. **behavioral**

This folder contains the behavioral data for subject UT014. The contents of each subject folder are grouped by the experimental paradigm. This subject participated in 'acc_reward', 'FR1', 'PS', 'pyFR_stim', and 'YC1'. A brief description of the various paradigms is outlined in the *Experimental Paradigms* section of this document.

A. **FR1**

This folder contains all of the relevant behavioral files for subject UT014's free recall data.

- **events.mat**
  This file contains all of the events for subject UT014's FR1 data. Each row in the event corresponds to a single event within each FR1 session (e.g. the beginning of the session, a single word presentation, a spoken response from the subject). Each field of the events structure is outlined later in this section.
- **session_0**
  The events.mat file described above contains the events for all sessions combined. There are also individual session folders that contain the events only for one session. If you look inside the FR1 folder for UT014, you will see that there are 5 session folders (session_0, session_1, session_2, session_3 and session_4). This indicates that UT014 participated in 5 sessions of FR1. A full session of FR1 contains 25 lists of 12 words but a subject may also only complete a portion of a session.

ii. **eeg.noreref**

This folder typically contains the non-rereferenced EEG files for a subject. For this tutorial, we will instead be using the eeg.reref folder described below.

iii. **eeg.reref**

The eeg.reref folder contains the raw EEG files after being re-referenced to the weighted average of the activity at all other electrodes. For the purpose of this tutorial, you do not need to understand the specifics of this re-referencing. Each EEG filename is in the format of [Subject Code]_[Experimental Paradigm]_[session number]_[Date (DDMonthYY]_[Military Time].[channel number] or [Subject Code]_[Date (DDMonthYY]_[Military Time].[channel number].Therefore, the file UT014_02Sep15_1705.001 corresponds to the EEG file for subject UT014 collected on September 2, 2016 at 5:05 PM for electrode number 001. There is usually one EEG file for each session for each electrode (i.e. EEG_filename.001 to EEG_filename.128 for channels 1 to 128). For the purpose of the tutorial, a subset of the channels are included. For subject UT014, we included channels 001, 002, 003, 042, 043 and 044. These electrodes are all located in the hippocampus.

II **eeg_toolbox**

This folder contains a toolbox of MATLAB function that will be used throughout this tutorial to analyze the EEG signals and will be useful for any EEG analyses you perform in the future.

III **beh_toolbox**

This folder contains functions for behavioral analyses that may be used in future versions of this tutorial.

Description of Free Recall events.mat Fields

The events.mat file is a MATLAB structure containing information about every single event during the experiment.

- **subject:** this field contains the subject code that corresponds with the event in that row
- **session:** this field specifies the session number for the event in that row.
- **list:** this field specifies the list number within the corresponding session. If a subject participated in 3 sessions, there will be 3 list #1s. If the subject completed 1 full session and 2 partial sessions, there may only be 1 list #25. This field is -999 for events that are not a part of a list of words or the recalls corresponding to a list of words.
- **serialpos:** this field specifies the position of the word corresponding with that event in the corresponding list. The first word for list 1 will have a serial position of 1. The sixth word will have a serial position of 6. The eighth word of the eleventh list will have a serial position of 8 (and list 11).
- **type:** this field specifies what type of event the corresponding event was. The possible entries for this are below
  - B: the beginning of the EEG recording
  - SESS_START: the beginning of the session on the testing computer
  - COUNTDOWN_START: the beginning of the countdown from 10 to 0 that precedes each list
  - COUNTDOWN_END: the end of the countdown that precedes each list
  - PRACTICE_WORD: a word that was presented during the first list of each session which is a practice list
  - PRACTICE_DISTRACT_START: the beginning of the distractor math problems for the practice list
  - PRACTICE_DISTRACT_END: the end of the distractor math problems for the practice list
  - PRACTICE_REC_START: the beginning of the 30 second recall period for the practice list
  - PRACTICE_REC_EN: the end of the distractor math problems for the practice list
  - TRIAL: this precedes any events for a new list
  - ORIENT: this corresponds to a fixation point on the screen (+) that precedes the list of words after the countdown.
  - WORD: the presentation of a word on a test list (these are the events to consider when doing an analysis on encoding).
  - REC_WORD: a vocalized response of a word during the recall period. These entries are manually scored by research assistants in the lab.
  - SESS_END: marks the end of the session
  - E: marks the end of the recordings
- **item**: this field specifies the specific word that was presented or recalled. This field contains an X for entries that do not correspond to a presented or recalled word.
- **itemno**: this field specifies the number of the word presented or recalled from the wordpool. Analyses can be performed using the word number or the actual word entry in 'item'.
- **recalled**: this field contains -999 for any entry that is not a word presentation. It has a value of 1 for words that were presented and remembered and a value of 0 for words that were presented and not remembered.
- **intrusion**: was the item from the current list (intrusion = 0) or from another list (intrusion = 1)
- **eegfile**: this field points to the EEG file that corresponds with that entry in the events structure. You will need to change this to point to the location of the EEG files where you saved the folder. The filename should be the same but the folder location will need to change.
- **eegoffset**: this field specifies the number of milliseconds from the beginning of the EEG file where the event in that entry begins. This is used by functions in the eeg_toolbox to determine what segment of the EEG data to analyze for a specific event.

Exercises

The exercises below are organized in order of increasing complexity. It may be helpful to read through each each exercise before attempting to complete it so that you have an idea of what variables need to be saved. Because you will be building on skills used in the previous exercise, the instructions for each exercise will contain slightly less detail. If you are stuck, try to use instructions from earlier exercises to determine how to move forward. The suggestions about how to complete the exercise are only suggestions. They are intended to point you in the direction of the easiest path to the answer but if you have another method you would like to try, you are encouraged to do so.

Optional: If you already have some experience working in MATLAB, consider making this exercise more challenging by reading through all of the exercises beforehand to write a single code that can be used to dynamically complete all of these analyses. While this is not necessary, writing a single code to perform any of the analyses below will be a good lesson in how to efficiently code analyses you will perform in the future.

Consider a practical example of how this can be accomplished: For the recall probability exercise below, you will first calculate the recall probability for 1 list for 1 subject. You will then calculate the recall probability for the entire session, across all sessions, and finally across all subjects and sessions. To create dynamic code that can accomplish all of these exercises, you can have a function with an input variable that specifies what grouping to perform the analysis on.

For example, entering 'list' will output the recall probability for each list. You can then index the probability for subject 1, list 1 to get the answer for exercise one. Entering 'session' will give you the recall probability for every session for all subjects. You can then index subject 1, session 0 for the second exercise.

Another way to increase the difficulty of these exercises is to try to accomplish them using as few 'for-loops' as possible. For-loops tend to be more computationally demanding than using vectors.

Behavioral Analyses

Calculating Recall Probability

In this first exercise, you will perform a simple behavioral analysis of free recall data. The probability of recall is simply the percentage of words that a subject remembered. You will first perform this analysis for one list for one subject. Once you are able to accomplish that, you will perform the analysis across all lists for one session for one subject. By the end of this exercise, you will be able to calculate the recall probability for across all sessions and all subjects. The steps necessary to perform these analyses are outlined below. As the exercises require you to calculate multiple recall probability, be sure to save the values out in an intuitive way that you can then work with later to average within a subject and then across all subjects.

- **Probability of Recall for UT014 Session 0 List 1**
  The objective of this exercise is to familiarize you with the events.mat file used to filter experimental events. In this exercise, you will easily be able to check that your code is running properly by manually calculating the recall probability for a list and verifying that your manual calculation gives you the same result as your code.

  1. Load the events.mat file located at LegaLabTutorial/sampleData/UT014/behavioral/FR1/events.mat
  2. Extract the events for session 0 list 1 that have type 'WORD'
  3. Using the 'recalled' field in the events structure, find the total number of words that were recalled
  4. Count the total number of words that were presented for session 0 list 1

5. To calculate the probability of recall, divide the number of word recalled by the number of words presented
6. Save out the probability of recall for R1134T session 0 list 1

- **Probability of Recall for UT014 Session 0 All Lists**
  This exercise should not require any skills beyond the skills used in the previous exercise. Because this exercise is over a larger set of data, it will be more difficult to manually check your work. This exercise should begin demonstrating why analyzing data is much more efficient using MATLAB.

  1. Load the events.mat file located at LegaLabTutorial/sampleData/UT014/behavioral/FR1/events.mat
  2. Extract the events for session 0 that have type 'WORD'
  3. Using the 'recalled' field in the events structure, find the total number of words that were recalled across all events for session 0
  4. Count the total number of words that were presented for session 0
  5. To calculate the probability of recall, divide the number of word recalled by the number of words presented during this session
  6. Save out the probability of recall for UT014 session 0.

- **Probability of Recall for UT014 All Sessions, All Lists** While there are many different ways to perform this analysis, one way to complete this exercise is using a 'for-loop'.You will want your code to check for the number of sessions and calculate a probability of recall for each session if a subject participated in more than one session. This exercise will also introduce you to plotting in MATLAB and calculating standard error of the mean.

  1. Load the events.mat file located at LegaLabTutorial/sampleData/UT014/behavioral/FR1/events.mat
  2. Determine the number of sessions that R1158T participated in based on the unique entries in the 'session' field
  3. Loop through each session number
  4. Extract the events for the session you are looping for that have type 'WORD'
  5. Using the 'recalled' field in the events structure, find the total number of words that were recalled across all events for the session you are looping through.
  6. Count the total number of words that were presented for that session.
  7. To calculate the probability of recall, divide the number of word recalled by the number of words presented during that session
  8. Repeat until you have a recall probability for all sessions for UT014
  9. Create a labeled and titled barplot showing the recall probability for UT014
     - The bar height should be the average of the recall probabilities for all sessions. You should also added errorbars to the barplot. The errorbar length should be 1 standard error of the mean (SEM). To calculate SEM, calculate the standard deviation of the recall probabilities across the sessions and divide the standard deviation by the sqrt(# sessions - 1). Because you cannot calculate a STD from 1 value and some subjects only participated in 1 session, it will be helpful to write your code so that the STD is only computed if there is more than 1 session.

- **Probability of Recall for All Subjects, All Sessions and All Lists** This next exercise will require you to navigate through folders using MATLAB. One approach is to use the 'dir' function to get all of the folder names in the data folder. These will be the subject names. You will then loop through

each subject, load the events structure for that subject and calculate the probability of recall for each session that subject participated in. This will require more sophisticated use of strings to generate the unique filenames for each subject. This exercise will also introduce a new method of plotting data: the grouped bar plot.

1. Write code to dynamically find the subjects who have FR1 events.mat files. Your code should be able to work if a new subject is added to the folder without you having to change anything in your code.
2. Load the events.mat file located at LegaLabTutorial/sampleData/[Subject]/behavioral/FR1/events.mat for each subject
3. Determine the number of sessions that each subject participated in based on the unique entries in the 'session' field
4. Loop through each session number
5. Extract the events for the session you are looping for that have type 'WORD'
6. Using the 'recalled' field in the events structure, find the total number of words that were recalled across all events for the session you are looping through.
7. Count the total number of words that were presented for that session.
8. To calculate the probability of recall, divide the number of words recalled by the number of words presented during that session
9. Repeat until you have a recall probability for all sessions for all subjects
10. Create a grouped barplot where there is one bar for each session for each subject. All sessions for a subject should be grouped together. Subject name will be on the x-axis and recall probability will be on the y-axis. There will be a separate bar for each session for each subject.
    - Because you are plotting the recall probability for each session, you do not need errorbars for this figure
    - Your plot should dynamically create the labels and groupings so that if another subject or session is added, the plot can be generated without having to change your code.
11. Plot a barplot showing the average recall probability across all subjects. First, average the recall probability for each subject across all sessions. Then, compute the SEM and mean of the average recall probability. Plot errorbars for the average recall probability across subjects. In your title, include the number of subjects that contributed to the average.

Generating Serial Position Curve

This exercise will guide you through how to generate a serial position curve (SPC). Simply put, the SPC is the recall probability at each item output position. In our experiment with 12 words per list, the SPC will have 12 recall probabilities. The first recall probability is the probability of a subject remembering the first word in a list, the second recall probability is the probability of a subject remembering the second item in a list and so forth. Whereas we were able to calculate the recall probability for a single list in the previous exercise, the SPC can only be calculated across lists because each list only has 1 occurrence of each item (i.e. 1 first item, 1 second item, etc.)

When plotted as a line, the SPC shows how likely subjects were to remember a word at each position in the list. Two common effects seen in the SPC are the primacy effect and the recency effect. The primacy effect refers to the increased probability of remembering items at the beginning of the list. The recency effect refers to the increased probability of remember items at the end of the list (more recent). These two effects are common but not always present for all subjects. The average serial position curve across all subjects,

however, should show some signs of this effect. The serial position curve will then have a U-like shape (higher probability of recall at the beginning and end of the list relative to the middle).
This exercise will also introduce you to plotting a line in MATLAB.

- **Serial Position Curve for UT014 Session 0**
  1. Load the events.mat file located at LegaLabTutorial/sampleData/UT014/behavioral/FR1/events.mat
  2. Filter for events for session 0
  3. Using the serialpos field, count the number of words that were presented for each condition
     - Hint: If a session ended in the middle of the list, this may not be the same as the number of lists.
  4. For each serial position, using the 'recalled' field, find the number of times that the word in that serial position was recalled.
  5. Calculate the percent of words at each serial position that were recalled by this subject for this session. Your result should have one probability for each serial position.
  6. Plot the serial position for UT014 Session 0 as a line. Title and label your plot accordingly.
- **Serial Position Curve for UT014 All Sessions**
  1. Load the events.mat file located at LegaLabTutorial/sampleData/UT014/behavioral/FR1/events.mat
  2. Loop through each session calculating the recall probability at each serial position for each session
  3. Calculate the SEM of recall probability at each serial position and plot the serial position curve with errorbars for this subject.
- **Serial Position Curve for All Subjects, All Sessions**
  1. Loop through each subject loading the events.mat file for each subject
  2. Loop through each session calculating the SPC for each session
  3. Average the SPC across sessions for each subject
  4. Plot the average SPC across subjects with SEM error bars. Include the number of subjects in the title of the plot.

Generating a Lag-Conditional Response Probability (CRP) Curve

This last behavioral analysis in this tutorial is arguably one of the most difficult behavioral analyses to perform using a free-recall dataset. The lag-conditional response probability (CRP) is a measure of how a subject moves from one response to another.

For example, if they remember item 3, what item do they remember next? If they remember item 4, their response has a lag of + 1 (moving from item 3, +1, to item 4). If they remember item 1, their response has a lag of -2 (moving from item 3, -2, to 1). A negative lag means the subject "moved" backwards as they remembered an item. A positive lag means they moved forward, following the order the items were presented.

Unlike the earlier analyses, the lag-CRP is not easily extracted from the event.mat structure. However, the events structure has all of the information you will need to compute it. This exercise will teach you how to creatively use the fields in the events structure to make inferences about free recall data.

To convert the response lag to a probability, you will need to calculate the number of times a subject could have possibly had a particular response lag and the number of times the subject actually produced a response

with that lag. The possible response lags depend on the item that was remembered and the items that have already been remembered.

For example, if the first item a subject remembers is item 2, it is possible to have a response lag of -1 (to item 1), +1 to item 3, +2 to item 4 all the way until +10 to item 12 (the last item in the list). This subject cannot, however, have a response lag of -2 (transitioning from item 2, -2, would result in a serial position of 0. The first word in a list has a serial position of 1). The subject also cannot have a lag of +11 (transitioning from item 2, +11, to item 13 would result in a serial position of 13 which is greater than the number of items on the list).

If after remembering item 2, the subject remembers item 8, the response lag is +6. Now, the subject can have lags from -7 (to item 1) through +4 (to item 12). However, because the subject has already remembered item 2, a lag of -6 (from item 8 back to item 2) is not possible. Subjects tend to remember items that were presented in close proximity to each other. The lag-CRP curve usually shows a high probability of making a -1 or +1 transition with the probability of larger lags becoming smaller (i.e. subjects rarely remember item 2 and then item 10 with a response lag of + 8).

Hopefully this example has shown you how complex the calculation of a lag-CRP can become and pointed out some of the considerations that go into computing this value. Once you are able to complete this exercise, the rest of this tutorial should be relatively easy to complete. Performing electrophysiological analyses will require minimally more advanced programming skills. After this exercise, the focus of this tutorial will shift from computer programming to neural signal processing.

- **Lag-CRP for UT014 Session 0, List 1**

  1. Load the events.mat file located at LegaLabTutorial/sampleData/UT014/behavioral/FR1/events.mat
  2. Filter for events for session 0, list 1
  3. You will now need to use events of type 'REC_WORD' to determine what word a subject said and then determine what order that word was presented in.
  4. You can start by filtering for events that are of the type 'REC_WORD' and have an 'itemno' that is greater than 0. You also want to make sure that the item is not an intrusion (i.e. 'intrusion' should be 0).
  5. For each 'REC_WORD', determine what serial position the word was presented at. You can do this by comparing the 'itemno' with the 'itemno' of the 'WORD' events for that same list. If you prefer to work using strings, you can also directly compare the item to the items presented during that list.
  6. For each recalled word, determine the possible lags. Possible lags are (1 - serial position of the recalled word) to (list length - serial position of the recalled word). These correspond to lags into the first item in the list and the last item in the list, respectively. A lag of 0 is not possible. A lag into a word that has already been recalled is also not possible.
     - Hint: for UT014, session 0, list 1, the first recalled word was 'GLASS' (serial pos: 1). The possible lags are [+1,+2,+3,+4,+5,+6,+7,+8,+9,+10,+11].
  7. Calculate the actual lag of the response.
     - Hint: for UT014, session 0, list 1, the first recalled word was 'GLASS' (serial pos: 1) and the second recalled word was 'STREET' (serial pos: 2). This response has a lag of +1.
  8. Determine the possible lags and actual lags for all recalled words for session 0 list 1 for UT014.

   - Hint: for UT014, session 0, list 1, the second recalled word was 'STREET' (serial pos: 2). Because 'GLASS' (serial pos: 1) has already been recalled, the possible lags into the third response are [+1,+2,+3,+4,+5,+6,+7,+8,+9,+10]. The following response was 'NAIL' (serial pos: 6) which has a lag of +4 from the word 'STREET'. Now, the subject has recalled items 1, 2 and 6. The possible lags from item 6 are [-3,-2,-1,+1,+2,+3,+4,+5,+6]. The next response was 'STAIR' (serial pos: 3). The actual lag from 'NAIL' to 'STAIR' is -3.

9. To generate the lag-CRP curve divide the number of times that each transition was possible by the number of times that transition was actually made.
   - Hint: Although each serial position only occurs once, each lag may be possible more than 1 time for a given list. In the example above, a lag of +1 was possible from both 'STREET' and 'GLASS'. Considering these two words (and 'NAIL' which followed) the subject could have made the +1 transition twice but only made it once. Therefore, the CRP of +1 is 50% (0.5).
   - Hint: To determine the possible lags, you must consider the words that have already been recalled, the distance from the last recall to the beginning of the list and the distance from the last recall to the end of the list. It may be helpful to keep track of all possible lags simultaneously. The maximum negative lag would be if the last item was recalled first and then the subject recalled the first item. This lag would be from serial position 12 to 1 or a lag of -11. The maximum positive lag would be from serial position 1 to serial position 12 (a lag of +11). A lag of 0 is not possible. There are thus 22 possible lags (-11 to 11).
10. The lag-CRP for 1 list may look unusual but it will be easy for you to check the accuracy of your code. Plot the lag-CRPs for lags of -3, -2, -1, 1, 2 and 3 for UT014, session 0, list 1 and verify that these values make sense based on the entries in the events.mat structure.

- **Lag-CRP for UT014 Session 0, All Lists**
  1. Load the events.mat file located at LegaLabTutorial/sampleData/UT014/behavioral/FR1/events.mat
  2. Repeat the exercise above but consider all responses across all lists.
  3. Plot the lag-CRP for the entire session as you did above
- **Lag-CRP for UT014 All Sessions**
  1. Load the events.mat file located at LegaLabTutorial/sampleData/UT014/behavioral/FR1/events.mat
  2. Repeat the exercise above looping through sessions. Calculate a separate lag-CRP for each session.
  3. Reproduce the CRP plot above for the average across sessions with SEM errorbars.
- **Lag-CRP for All Subjects, All Sessions**
  1. Load the events.mat file for each subject
  2. Repeat the exercise above for all subjects
  3. Average the CRP across sessions for each subject.
  4. Reproduce the lag-CRP plot for the average across subjects showing SEM errorbars.

Electrophysiological Analyses

Now that you are familiar with how to utilize the event.mat structure, you can begin analyzing the EEG data collected during the behavioral task. Before you can perform the analyses below, you will need to modify the 'eegfile' field for all of the event.mat structures to point to the location where you have the data saved. The default filename begins with '/project/shared/lega_ansir/subjFiles/[subjName]/eeg.reref/[EEG filename]'. You

will want to replace '/project/shared/lega_ansir/subjFiles' to the path where the 'LegaLabTutorial' folder is stored and add the folder 'sampleData'. For example, if the 'LegaLabTutorial' is saved in the Documents folder for a user 'jdoe', the EEG path for UT014 collected on September 2, 2015 should be changed from to '/Users/jdoe/Documents/LegaLabTutorial/sampleData/UT014/eeg.reref/UT014_02Sep15_1705'. To determine the path where you have your data saved, you can perform a Google search for 'How to find the path for a file on a Mac' or 'how to find the path for a file on a PC'. It is often easiest to check for file paths on a Mac using the terminal. A basic introduction to navigating your computer using terminal is inlcuded

Plotting Raw Voltage

This first exercise will introduce you to how EEG data is extracted for an electrode for a specified time window relative to an event. You will learn how to plot the voltage recorded from one electrode for a single event.

- **Voltage for UT0145 ch 1, 1 Event**
  1. Load the events.mat file for UT014
  2. Change the path in events.eegfile to point to the location of the EEG files on your computer.
  3. Extract the first event corresponding to word presentation (i.e. type 'WORD').
  4. calculate the voltage for that event beginning 150 ms before the word onset and ending 2000 ms after the word onset. Use a buffer of 1000 ms. Filter out line-noise using a [58 62] Hz first order 'stop' filter.
     - Hint: this requires using the EEG toolbox. Make sure the eeg_toolbox folder in 'LegaLabTutorial' is in your MATLAB path. You can Google how to add a folder to your path if you are unsure.
     - Hint: the function used to calculate voltage is 'gete_ms'. Open the function for some background on how to use it. You can reference the section on 'Using gete_ms' below for more details.
  5. Plot the voltage labeling the time relative to word onset on the x-axis. Indicate the word onset with a green vertical line and the word offset with a red vertical line.
  6. Label and title the plot appropriately specifying the subject, channel number, word presented and sampling rate.
- **Voltage for UT014 ch 1, 1 Event with Resampling**
  1. Calculate the voltage for the same event and time window from the exercise above but now resample the data to 100 Hz.
  2. Plot the resampled voltage on the same plot as the voltage calculated above to compare the two traces.
     - Hint: Because you now have a different sampling frequency, there will be a different number of data points for these two lines. You will need to find a way to align the two voltage traces so that the raw and resampled voltage at a single time point will overlap.

Plotting Event-related Potential (ERP)

In the previous exercise, you plotted the raw voltage recording from an electrode during a single event. A more interesting question we can ask is what voltage changes are evoked by a stimulus (e.g. studying a word). We can determine this by averaging the voltages recorded across many events. The resulting average is known as an event-related potential (ERP).

The idea behind an ERP is that any voltage deflection that persists after averaging across many events is a time-locked response to the event being evaluated. For example, if every time a subject studies a word, there is a positive voltage deflection, averaging across all encoding events will reveal a positive voltage deflection. However if there is a positive deflection during some words and a negative deflection during others, the averaged voltages will be roughly 0.

Similarly, if there is an early positive deflection on some words and a late deflection for other words, the average voltage (ERP) may show a broad positive deflection with a very small magnitude or may not show any voltage deflection at all. An ERP will only show strong effects for voltage changes with the same directionality and timing across many trials.

It is important to note that a voltage deflection in the ERP does not necessarily reflect a true voltage deflection that occurs across every encoding event. If a few trials show very large positive voltage deflections but a majority of trials show a very small negative voltage deflection, the average may appear to show a positive deflection (though it will not be as large as it was for the few trials that had a true positive deflection).

This exercise will introduce you to how you can compute an ERP for a single channel as well as how to evaluate the ERP across several channels. You will use the baseline re-referencing feature of gete_ms in this exercise.

- **Plotting the ERP for UT014, channel 1 All Encoding Events**
  1. Filter the events.mat structure for UT014 for only the encoding events.
  2. Calculate the raw voltage for all of these events using gete_ms
  3. Find the mean and standard deviation of voltage across all events
  4. Plot the ERP with errorbars for this channel. Label and title the plot accordingly.
- **Plotting the ERP for 2 Channels for All Encoding Events for UT014**
  1. Calculate the ERP for channel 2 as you did for channel 1 in the exercise above
  2. Plot the ERP with errorbars for channel 2 and 1 on the same figure labeling each channel.
- **Plotting the Baseline Re-referenced ERP for 2 Channels** You may have noticed that the ERPs for the two channels in the exercise above have different magnitudes. It is sometimes more appropriate to compare deflections from a baseline period than it is to compare the ERPs generated from the raw voltage. In this exercise, we will use the pre-stimulus period (time before the word is on the screen) as a baseline. This will allow us to evaluate whether seeing a word on a screen causes voltage to increase or decrease relative to the baseline period when the patient is theoretically in a resting state.
  1. Calculate the ERP for channels 1 and 2 as you did in the exercise above. This time, however, you will perform a baseline subtraction using the -150 ms to 0 ms baseline.
     - Hint: refer to the section on gete_ms for information on how to perform a baseline subtraction.
  2. Plot the baseline re-reference ERP for each channel and label your plot accordingly.
  3. Add a horizontal line along the x-axis at y=0. Clearly indicate the word onset and offset in your figure. Remember that the word is on the screen for 1600 ms.

  - **Bonus Exercise:** In this exercise you used the built-in functionality of gete_ms to baseline re-reference the voltage for both signals. Try to manually re-reference the signal by subtracting the average voltage during the baseline period from the voltage at each time point. Plot your result on top of the output from gete_ms. The lines should be exactly the same.

- **Plotting the Average ERP in the Hippocampus** The last exercise demonstrated how using a baseline to re-reference voltages recorded at two electrodes can help draw meaningful conclusions from effects recorded at different electrodes. We will now use this skill to generate normalized ERPs for all of our electrodes to generate an average ERP across all electrodes. The available EEG files for this tutorial are all in the hippocampus so this excercise will generate a figure displaying the ERP in the hippocampus.

  1. Determine what electrodes you have EEG files for for each subject.
     - Hint: The suffix of the files in the eeg.reref folder correspond to the electrode numbers that you have eeg files for. For example, you have EEG files for channel 1, 2, 3, 42, 43, 44 for UT014.
  2. Loop through each electrode for each subject and calculate the ERP for all encoding events.
     - **Challenge Exercise:** Because you will eventually be working with a dataset with hundreds of subjects, it is often useful to write code that can automatically read files to generate a list of available electrodes. Try to write your code so that it reads the contents of eeg.reref to determine the available electrodes. If your code is correct, adding an additional EEG file for a new channel will generate an ERP for that channel without you changing anything.
  3. Average the ERP across all electrodes for all subjects to generate an average ERP for the Hippocampus. Note that for this tutorial, you were only given EEG files for electrodes in the hippocampus.
  4. Plot the average hippocampal ERP with errorbars.

Experimental Paradigms

FR1: verbal free recall paradigm for the RAM project FR2: this is the same free recall task as FR1 but stimulation is delivered on random words within each list.
pyFR: free recall paradigm that is similar to FR1 with minor differences in list length, recall duration and number of lists.
pyFR_stim: this is the same free recall task as pyFR but stimulation is delivered during entire lists. For some subjects stimulation was delivered during the entire encoding period OR the entire recall period.
AccRew: this is a reward processing task
PS2/PS3: these are tasks where we deliver stimulation with varying parameters (amplitude, frequency, duration, location) without any behavioral tasks
catFR1: this is similar to FR1 but the words within each list fall into category groupings (e.g. Animals (dog, cat, fish), Fruits (apple, orange, pear), Tools (saw, drill, hammer)); catFR2 is the same task but with added stimulation
YC1: this is a spatial navigation task
YC2: the same spatial navigation task with stimulation
PAL1: a paired-associate task where subjects have to remember a word from a pair of words instead of just any word from a list.
PAL2: the same as PAL1 with added stimulation.

EEG Toolbox

This section will provide a brief overview of the most useful functions in the eeg_toolbox and how they can be used to analyze EEG signals.

Using gete_ms

gete_ms is the core function in the eeg_tooolbox that extracts voltages recorded from an electrode. The function uses the eegfile path in the event.mat structure to locate the eegfile that corresponds with an event and the eegoffset field in the events.mat file to know what time relative to the beginning of the recording to analyze. This function will only work if the event.mat structure points to a valid EEG file.

The 'channel' input of the function further specifies which eegfile to access. Inputting 21 for channel will append '21' to the eegfilepath and access the eegfile corresponding to channel 21. The 'LegaLabTutorial' folder does not contain the EEG files for all channels so you can only access the channels available in the eeg.reref folder for each subject. Once working in the lab, however, you will have access to the files for all of the channels used.

Inputs

- channel: the channel number you wish to analyze
- events: the events you want the voltages for. The output will give you a voltage for each time point for each event. The output matrix is described in more detail below.
- DurationMS: the length of time (in milliseconds) of EEG recording you would like to analyze
- OffsetMS: the time relative (in milliseconds) to the beginning of the event you would like to begin analyzing the EEG data
- BufferMS: the time (in milliseconds) before and after your desired timewindow to use as a buffer to prevent edge artifacts. The importance of this field is beyond the scope of this tutorial. You can use 1000 ms for this.
- filtfreq: the frequencies you want to filter out or allow through. It is typical to use [58 62] to filter out 60 Hz linenoise from electrical outlets in the U.S..
- filttype: the type of filter you want to use. 'stop' does not allow the specified frequencies through and is used for line-noise filtering. 'low' allows any frequencies below the
- 'filtfreq' through (low-pass). 'high' allows any frequencies above the specified frequency through (high-pass).
- filtorder the significance of this input is beyond the scope of this tutorial. You can use an input of 1 for this field.
- resampleFreq: this entry is used to change the sampling rate of the EEG file. This is useful if trying to compare EEG recordings for patients with different recording rates. For instance, if a subject at the University of Pennsylvania had a 500 Hz recording and subjects at UT Southwestern have 1000 Hz recording, you can resample all of the data to 500 Hz.
- RelativeMS: this input allows you to subtract a baseline period from the voltage. This is useful to mitigate the effects of voltage drift across a session and to enable a more standardized comparison of activity from different electrodes and subjects.

Outputs

- EEG: this is a matrix of voltage values with dimensions (# of events by # of times).
- resampleFreq: this is an output of the sampling rate used

Example

EEG = gete_ms(21,events,1800,-200,1000,[58 62],'stop',1,200,[-200 0]);

This will return a matrix with as many rows as number of events in 'events' and 360 columns. The resample frequency is 200 Hz so each sample is 5 ms. The duration of the voltage computed is 1800 ms (360 samples). The voltage output begins 200 ms before the onset of the word and ends 1600 ms after the word onset. Therefore, samples 1 to 40 are before the word, and samples 41 to 360 are after the word. A baseline subtraction of the average voltage from -200 ms to 0 ms was performed.

Exploring Folders on your Apple Computer Using Terminal

It is often much more efficient to work with folders and files on your computer from the command line (Terminal on an Apple Computer). Depending on your operating system, the command line interface on your computer may be different but the functionality should be preserved across all systems.

Terminal is effectively the same as 'Finder' on an Apple Computer or 'My Computer' on a PC but using it requires becoming acquainted with some common commands. While it may seem to add unnecessary complexity at first, you will soon find that navigating through your computer from the command line can help you accomplish seemingly tedious tasks much more efficiently.

As you first begin working in Terminal, it may be helpful to have a file browser window (such as Finder on a Mac) open. As you use terminal commands to navigate from your computer, verify that the result of navigating through your computer with the file browser is the same as navigating through the command line.

Common Command Line Functions

- **pwd:** lists current directory
- **ls:** lists contents of current directory
- **ls -l:** lists the read-write-execute permissions for each file in the current directory, the owner and group specifications for a file, the size of the file, the date it was created, and the file name.
- **cd [path]:** change directory to the path specified by [path]. For example:
  - **cd /Users/jdoe/Documents/** will move to the Documents folder for the user account 'jdoe'. Typing 'ls' after this command will show the contents of that folder.
  - **cd /Users/jdoe/Documents/exampleFolder** will move to the folder 'exampleFolder' if it exists. Again, typing ls will show the contents of this folder.
  - **cd exampleFolder** while already in Documents will yield the same result as typing the full path for exampleFolder. Typing the forward slash before the path allows you to change to a directory from anywhere on the computer (as long as there is only one folder with the name that follows the first forward slash). Excluding the forward slash before the path will change directories to files within the current folder.
  - **cd ./../** will move out of the current folder into the folder where it is contained.
  - **cd ~** will move to the root or home directory for the current user.
- **cp [source path] [destination path]:** will copy the file at [source path] to [destination path]. Entering '.' as destination path will copy the file at source path to the current directory.

- **cp -r [source path] [destination path]:** including the '-r' specification indicates a recursive copy which allows you to copy a folder and all of its contents rather than just a single file.
- **scp [source path] [destination path] and scp -r [source path] [destination path]:** These commands are the same as cp but they allow you to copy a file from a server. For example:
  - **scp -r jdoe@rhino2.psych.upenn.edu:/data/eeg/R1053T/behavioral/ /Users/jdoe/Documents/R105** will copy the folder on the rhino server at the specified path to the specified local path.
- **mv [source path] [destination path]:** moves the source file/folder to the destination.
- **rm [path] and rm -r [path]:** deletes the file or folder respectively located at the specified path.
- **ssh -XY [username]@[serverURL]:** prompts you to enter a password to connect to the specified server with the specified username.
  - **ssh -XY jdoe@rhino2.psych.upenn.edu** will prompt you to enter the password to connect to rhino2 under the username jdoe.