Lab 4

Overview

This lab will provide you with an introduction to fundamental concepts used in analyzing EEG data, along with an exploration of one of the widely acclaimed EEG-processing toolboxes frequently employed in research settings.

Deliverables

This lab is worth 8% of your overall grade. The grade will be assessed based on in-lab components (2%) and a lab report (6%) to be submitted on Quercus no later than November 6th at 11:59pm.

You **must** attend the session for the entire duration of the lab, unless you finish early and your TA confirms that you have completed the required deliverables and may leave.

Your grade will be assessed based on the following requirements:
- Demonstrating completion of each section to your TA
- Answering TA evaluation questions during the session

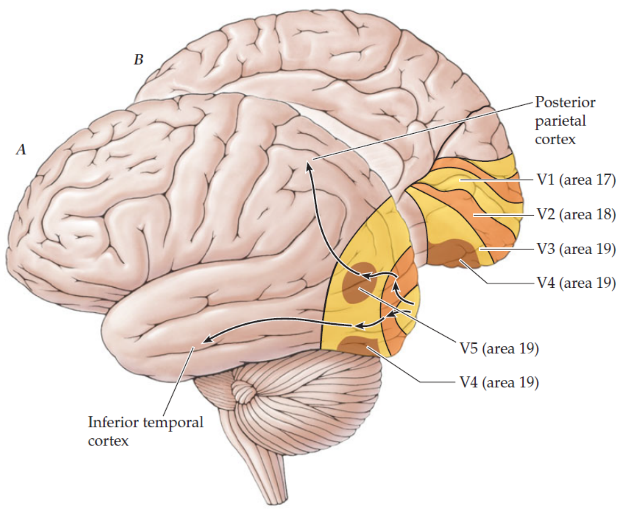
Preparation

Following, you will find key concepts that are necessary for your understanding of this laboratory.

Visual Cortex

The primary visual cortex, often referred to as V1, is a specialized region within the occipital lobe responsible for processing visual input originating from the eyes. V1 establishes intricate connections with other regions within the occipital lobe, notably higher-order visual areas (V2-V5). These connections facilitate the integration and processing of visual information.

Among these regions, the secondary visual cortex, or V2, assumes a prominent role. By relaying information to other higher-order areas, it can dissect various aspects of visual stimuli, such as motion, color differentiation, and form identification.



Martin, J. (2012). *Neuroanatomy Text and Atlas*

Subsequently, the visual information is transmitted to other brain regions, located in different lobes. This wider distribution allows for heightened integration, ultimately enabling the complex tasks of object recognition and precise object localization to be effectively performed.

Intermittent Photic Stimulation

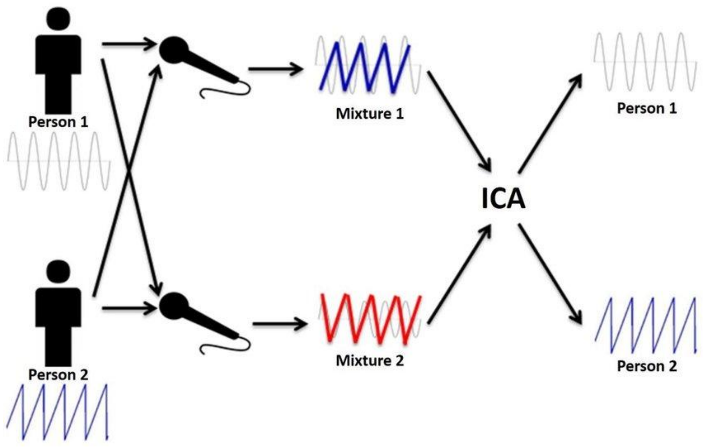
Intermittent Photic Stimulation (IPS) is a common procedure performed in the electroencephalography (EEG) laboratory in children and adults to detect abnormal epileptogenic sensitivity to flickering light (i.e., photosensitivity).

When IPS is applied, it can induce rhythmic electrical activity in the visual cortex, leading to prominent responses in the EEG channels overlying the [redacted] lobe.

Independent Component Analysis

Independent Component Analysis (ICA) is a source-separation method that aims to locate individual signals (components), assuming that its input is a linear combination of the amplitude of these individual components.

An illustrative scenario for ICA involves its application in isolating voices from an audio recording captured during a party. Consider a situation where two microphones are placed at distinct locations, recording a conversation between two individuals. The microphone signals are the combined output of both voices. In the case of stationary signal sources, the coefficients of this linear combination remain constant. By inferring these coefficients through a data regression, it becomes possible to invert the mixing process, effectively separating and isolating the distinct voices.



Lubo-Robles, D., & Marfurt, K. J. (2019). Independent component analysis for reservoir geomorphology and unsupervised seismic facies classification in the Taranaki Basin, New Zealand. *Interpretation, 7*(3), SE19-SE42.

ICA is commonly used in EEG to either isolate sources of noise and subtract the artifacts generated by these sources, or as a data reduction technique to analyze data solely based on component time series rather than electrode time series.

MNE Python

MNE stands as a valuable ally in the realm of neurophysiological exploration. As an open-source Python package, MNE allows for the processing of a wide range of human neurophysiological data in various modalities such as MEG, EEG, sEEG, ECoG, NIRS, and beyond.

For further information and resources, you can explore the official MNE documentation at the following link: https://mne.tools/stable/index.html

Experiment

During this experiment you will be handed 2 EEG recordings –in the form of text files– of a subject undergoing Intermittent Photic Stimulation.
The goal of the lab is to employ EEG analysis tools to determine the frequency at which the stimuli were presented.

Data analysis

For the data analysis of this laboratory Jupyter Notebooks is recommended, as it allows to perform this analysis in separate steps; but any IDE that supports Python is allowed.

Packages

For this laboratory you will need the following packages:

- Pandas
- Numpy
- Matplotlib
- MNE

Load the .txt file as pandas DataFrame

1. Load the following OpenBCI files: Rec A & Rec B to pandas using the *pd.read_csv* function and save it into a variable of your choosing. Please consider the following points regarding optional arguments:
    a. Because of the way that OpenBCI generates recordings as text files it is important to skip the first 4 rows of information. Set the optional argument as "**header=4**"
    b. The "Sample Index" column, because of parcellation issues, must not be used as the index of the DataFrame. Set the optional argument as "**index_col=False**"
    c. The recording generates much more data than just EEG recordings; such as accelerometer information, analog recordings, timestamps, etc. Since the only channels of interest are located in columns 2 through 9, set the optional argument as "**usecols=range(1,9)**"
2. Set the name of the pandas DataFrame's columns as **['FP1', 'FP2', 'C3', 'C4', 'P7', 'P8', 'O1', 'O2']** in that specific order.
    a. There's many ways to doing this, we recommend using the following method: *DataFrame.columns = [list of names]*
    b. Make sure that there's no typos and everything is in uppercase, as it is MNE's preferred notation.

Convert the DataFrame to a numpy array and create an MNE RawArray

1. Before loading the data into MNE you have to make sure it is in the correct format.
    a. MNE employs the convention of "channels x samples" as opposed to OpenBCI's "samples x channels". For this, it is necessary to transpose the data frame.
    b. MNE's voltage convention is to measure in Volts (V), whereas OpenBCI records in microVolts (μV). Therefore, you must convert to the appropriate units. Multiplying the full data frame by **1e-6** is the recommended method.
2. Now that the data is ready for MNE, you will create an MNE array and its underlying information.

    a. Using the *mne.create_info* function, create an info instance.
        i. The first argument, *ch_names,* must be a list of strings containing the channels names. These channel names must be exactly the same as those that were used to name the columns of the pandas data frame.
        ii. The sampling frequency, *sfreq*, must be the one specified by the OpenBCI recording. You can find it in the 3rd row of the recording text file.
        iii. The last argument, *ch_types*, must be a specification of which type of recording was performed. It must be a list of strings, where each string represents each channel's type. Since all 8 channels are EEG signals, this argument must be a list containing the string "eeg" eight times.
    b. Using the *mne.io.RawArray* function, create a raw object and save it into a variable of your choosing. This function requires two arguments.
        i. The first argument is the data itself (the pandas dataframe)
        ii. The second argument is the info instance you just created.
3. Lastly, it is necessary to also include the 3D coordinates of each electrode to properly locate where the signals originate from in space.
    a. Read the montage coordinates from the file *electrode_positions_8channel.sfp* using the *read_custom_montage* function and save it to a variable.
    b. Use the *set_montage* method from the raw object to incorporate the montage by using the variable you just created in the last step as an argument.

Apply bandpass filter from 0.1 to 100 Hz

1. Visualize the data using the *plot* method from the raw object. The following arguments will be needed.
    a. To see the full 25 seconds of recording, set "**duration=25**".
    b. To observe the complete amplitude of the signals, set "**scalings=dict(eeg=200e-6)**"; this argument allows us to plot signals that are 400μV peak-to-peak.
    c. Some of the signals are larger than the set scaling, to see the full signal set "**clipping=None**".
2. The signals should look like thick traces, rather than smooth lines; this is common because of high frequency noise as well as power line noise. As well, some channels slowly change voltages until they overlap others, a problem caused by slow drift. To eliminate these problems filtering is needed.
    a. Apply a **bandpass filter from 0.1Hz to 100Hz** employing the *filter* method from the raw object. Only the first two arguments (low pass-band and high pass-band) are required.
    b. Apply a **notch filter at 60Hz** using the *notch_filter* method on the raw object. Only the frequency to filter out from data is necessary to specify.

Preliminary plotting and rejection of bad channels

1. Plot the signals again, as specified in the previous section. Compare how different the signals look.

2. Even after filtering, you may realize that the information from some channels is of bad quality; channels that appear flat, or have high variance should be rejected.
    a. In this specific case we know that both central electrodes “**['C3','C4']**” yielded low quality data.
    b. The info structure of the raw object must be edited to inform from which electrodes further analysis should not be included. The next command can be used: ***raw*.info["bads"].extend(*badChanns*)**
        i. Where *raw* is the name of the raw object, and *badChanns* is a list of strings containing the names of the electrodes to discard.
3. After rejecting the channels, plot the signals again. The rejected channels should appear grayed out.

Plot the spectral response for each channel

1. Employing the *compute_psd* method of the raw object, generate the spectral analysis of each electrode.
    a. For illustrative purposes, set the minimum frequency as 1Hz and the maximum as 30Hz, since the frequency at which the test subject was exposed to was in between this range.
    b. Plot the power spectrum of the recording by adding the *plot* method at the end of your command as follows: ***raw*.compute_psd(*arguments*).plot()**
        i. Where *raw* is the name of the raw object, and *arguments* are the minimum and maximum frequencies to calculate.
2. Can you identify the frequency of the stimulus at which the subject was exposed to?

Create the ICA object and fit it to the data

1. Create an ICA object using the mne.preprocessing.ICA function and save it into a variable of your choosing.
    a. No arguments are needed since the default settings are appropriate.
2. Use the fit method of the ICA object to run the ICA decomposition on the data.
    a. The only necessary argument is the name of the variable where the raw object is stored.

Visual inspection of components and frequency response

1. Using the plot_properties method of the ICA object, plot the properties of each of the detected components. Three arguments will be necessary:
    a. The first component is the instance of the raw object.
    b. The method will automatically show the first 5 components. To show all 6 components, the pick argument should be set as follows: “**picks=range(6)**”
    c. To manually set the limits of the x axis of the power spectrum graph the next argument should be set: “**psd_args=dict(fmin=1, fmax=30)**”
2. Based on the peaks of the power spectrums of each component, can you guess at which frequency the subject was being stimulated?
    a. Two tips/questions that may help you answer this question are the following:

i. The components are ordered from most variance to least; therefore, it is common for the first few components to represent noise or artifacts (such as blinks), and the last components to hold little significance.
ii. Components are accompanied by maps of where they are most prominent during the recording. In which electrodes would a visual stimulus be most prominent?

References

Kasteleijn-Nolst Trenité, D., Rubboli, G., Hirsch, E., Martins da Silva, A., Seri, S., Wilkins, A., Parra, J., Covanis, A., Elia, M., Capovilla, G., Stephani, U., & Harding, G. (2011). Methodology of photic stimulation revisited: Updated European algorithm for visual stimulation in the EEG Laboratory. *Epilepsia*, *53*(1), 16–24. https://doi.org/10.1111/j.1528-1167.2011.03319.x

Lubo-Robles, D., & Marfurt, K. J. (2019). Independent component analysis for reservoir geomorphology and unsupervised seismic facies classification in the Taranaki Basin, New Zealand. Interpretation, 7(3), SE19-SE42.

Martin, J. H. (2012). Neuroanatomy Text and Atlas (4th ed.). McGraw Hill. p. 171.