Welcome to AMPLab!

Hi, this is a practice project for you to get familiar with deriving click-evoked auditory brainstem response (ABR) using python from raw EEG data.

We will go through the analysis step-by-step.

Set up Python3 environment

1. Download Anaconda3
2. Choose your IDE (e.g., Spyder, VScode, PyCharm, etc).
3. Install basic packages including:

- Signal processing: `numpy`, `scipy`
- Data visualization: `matplotlib`
- EEG signal processing: `mne`
- (may or maynot be needed in the analysis) Audio-visual experiment package: `expyfun`

p.s. The last two may need some dependencies.

Dataset description

Click waveform dataset

Clicks are generated from Poisson process with a rate of 40/s. There are ten trials of click data that are presented, each contains 1-min long clicks.

Source: https://rochester.box.com/s/uhc3tntssfdj2d8w97sjflmmy7zj0rzu

The dataset includes click trial from click000.wav to click009.wav, which are presented in order.

The sampling frequency of the waveform is **48000 Hz**.

EEG dataset

Dataset for subject from "subejcts001" to "subject024", excluding "subject014" and "subject021" because of bad quality.

```
subject_list = ['subject001', 'subject002', 'subject003','subject004' ,
                'subject005', 'subject006', 'subject007', 'subject008',
                'subject009', 'subject010', 'subject011', 'subject012',
                'subject013', 'subject015', 'subject016', 'subject017',
                'subject018', 'subject019', 'subject020', 'subject022',
                'subject023', 'subject024']
```

For this particular EEG dataset, the data format is `.fif` (e.g., subject001_click_eeg.fif). However, the raw dataset from BrainVision will include three files: `.eeg`, `.vmrk` and `.vhdr`.

There are two channels in the dataset `['EP1','EP2']`. The sampling frequency of the EEG data is **10000 Hz**.

EEG data processing

It is highly recommended to take a look at the tutorials of EEG analysis on `mne` website: https://mne.tools/stable/auto_tutorials/index.html

Load EEG data

Use `mne.io.read_raw_fif()` for `.fif` format file.

For brainvision data, use `mne.io.read_raw_brainvision()`, and the argument should be the `.vhdr` pathway. Note, your `.eeg`, `.vhdr` and `.vmrk` files should be in the same folder directory.

```
eeg_raw = mne.io.read_raw_fif(eeg_fif_path, preload=True)
```

or

```
eeg_raw = mne.io.read_raw_brainvision(eeg_vhdr_path, preload=True)
```

Preprocessing

`scipy` is used for filtering.

1. High-pass the EEG at 0.1 Hz or 1 Hz to filter out slow drifts.
2. Notch filter at 60, 180, and 540 Hz to filter out power line noise. The three frequencies used here is for this specific EEG dataset. For your own data, you can use `eeg_raw.plot_psd()` function to plot the psd of the raw EEG, and it is easy to spot the noise.

Epoching

Epoching is used to extract chunks of EEG signal that time-locked to your stimulus. For this dataset, we need to extract the 10 epochs of 1-min long click trials. Every epoch is annotated with the trigger marker at the beginning. For this data, it is marked with a '1' followed by a 10-digit '4's and '8's which is coded by the stimulus type and trial number. They are coded accoding to this function:

```
trigger = [(b + 1) * 4 for b in decimals_to_binary([0, cn], [n_bits_type,
n_bits_epoch])]
```

For example, click is type 0, and for `click009.wav` trial, the binary code is `[0, 0, 0, 0, 0, 0, 1, 0, 0, 1]`, then is converted to trigger `[4, 4, 4, 4, 4, 4, 8, 4, 4, 8]`.

Since the clicks trials are presented in order here, we don't need to worry about this coding too much. But it is useful for randomized trials.

Use `mne.events_from_annotations()` to get the trigger list, and then use `mne.Epochs()` to get the click epochs data.

Analysis

Convert click waveforms to pulse trains

To do the cross-correlation, we need to convert the click waveforms to a pulse train.

1. Read the click waveform files. You can do it with `expyfun.io.read_wav()`.
2. Rectify the waveform by getting the absolute value.
3. Identify the time point when the waveform has a click happen, and mark that time point with a pulse (a number '1' in the sequence).

Note, the output pulse train should match the sampling frequency of EEG data in order to do the cross-correlation.

Fourier Transform

We usually complete the cross-correlation in frequency domain. (You could also do it in time domain, but frequency domain multiplication is more efficient.)

There are originally two channels of EEG data, average the two channels first.

Use `numpy.fft.fft()` for fast fourier transform of EEG and click pulse train.

Cross-Correlation in Frequency Domain

The cross-correlation in time domain is equivalent to the multiplication of FFT(EEG) and the conjugate of FFT(pulse_train) in frequency domain. Remember, we still need the ABR result to be in time domain, so use `numpy.fft.ifft()` to inverse the fourier transform.

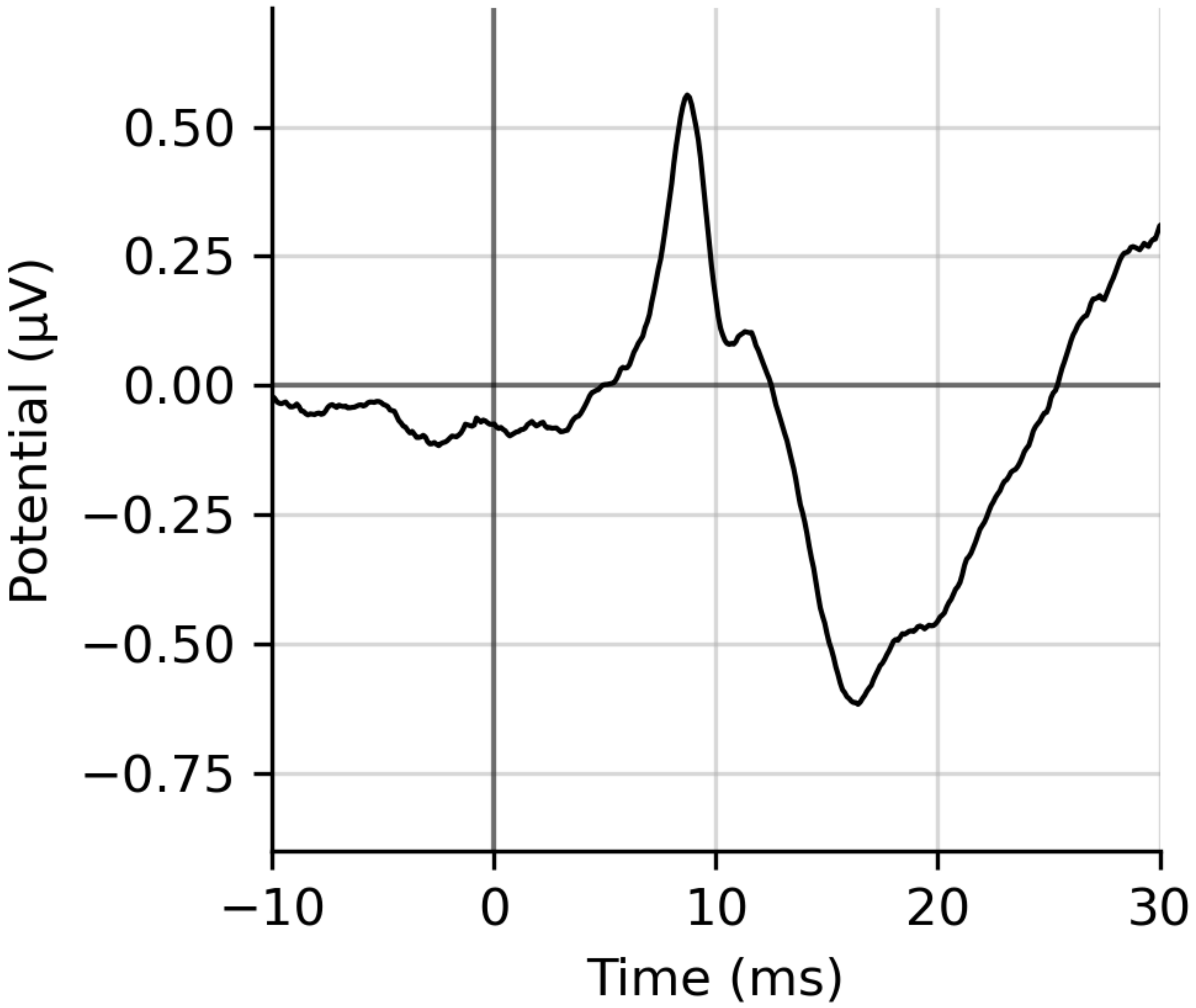
Get the Averaged Click ABR

We now have the click-evoked ABR for each trial. But what we need is the averaged ABR for that subject. Therefore, we need take the mean of the 10 trials.

The ABR we get is the sum of click ABR through the trial, which is actually (40 clicks/s * 60 s) times the ABR for a single click. So we then need to devide the amplitude of the ABR by (40*60).

To show the click-evoked ABR, we usually plot the time range [-10, 30] ms. Since the derived ABR is circular, we can concatenate the last 10 ms and the first 30 ms.

The final click-ABR should looks like this figure:


Potential (µV)
0.50
0.25
0.00
−0.25
−0.50
−0.75
−10
0
10
20
30
Time (ms)