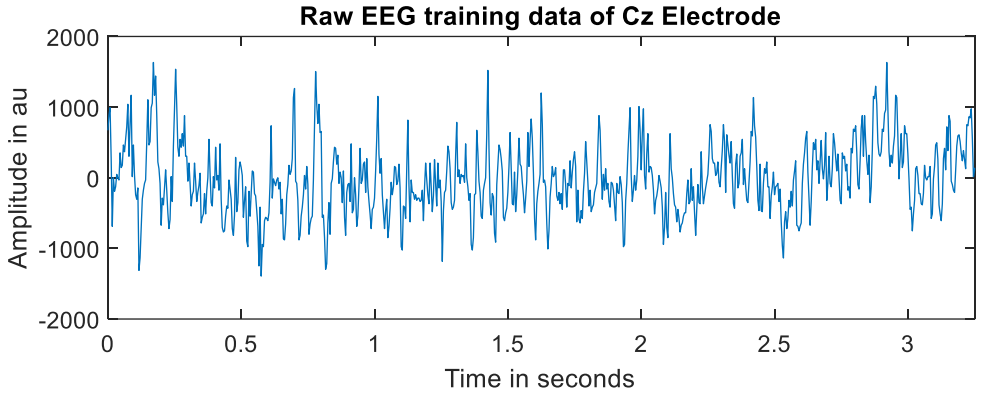
Name: Kamran Equbal

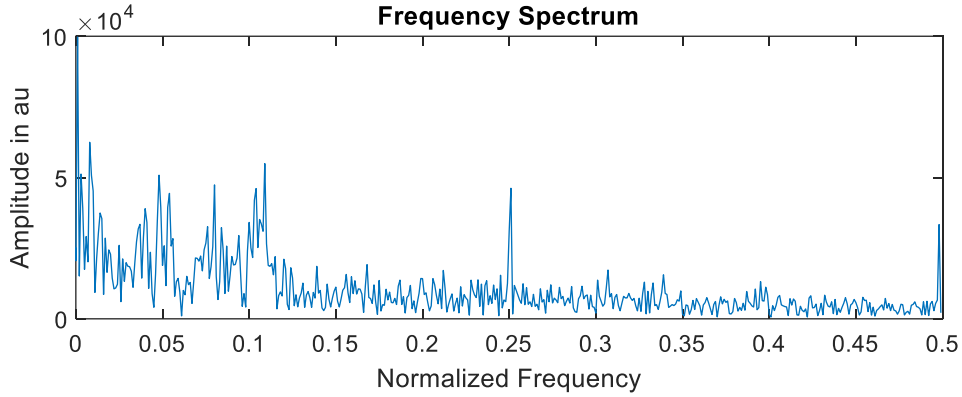
In this project from the raw recorded EEG data for a specific stimulus for P300 wave, we pick up clean and then train a classifier for that particular stimulus P300 wave.

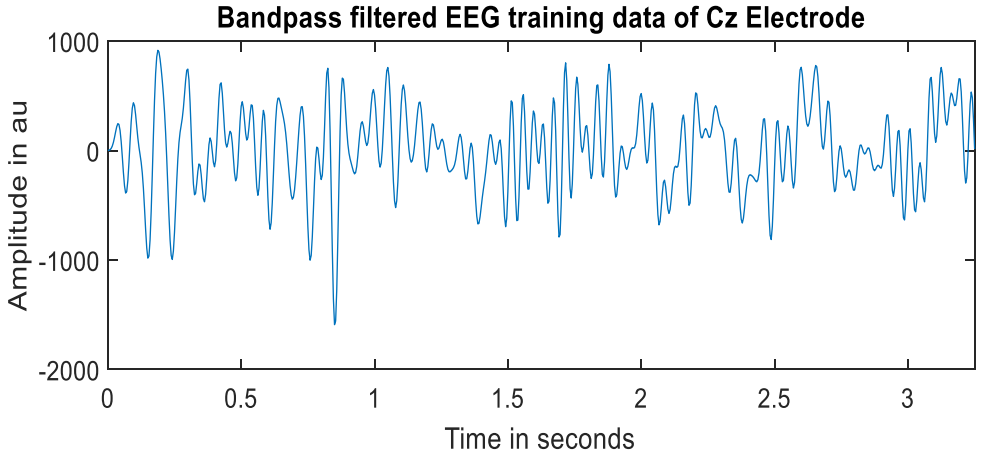
Instructions for running: Place all the attached files into Matlab directory and then run Project1b.m

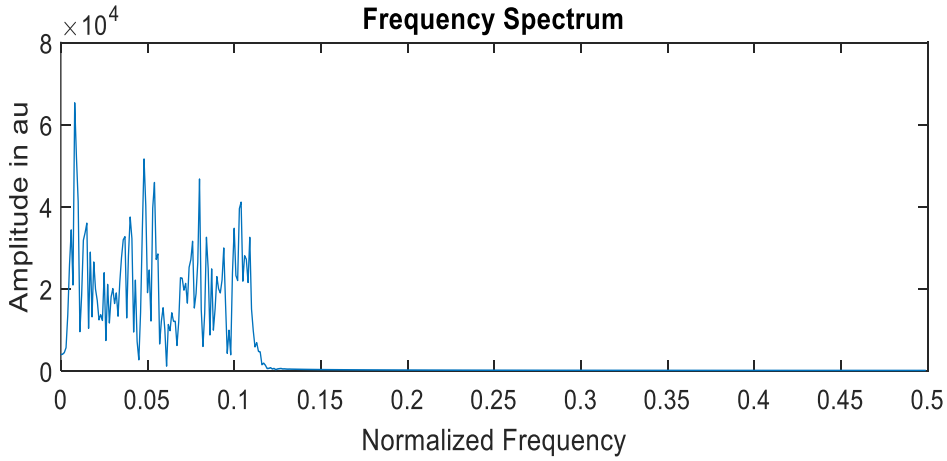
1) **Select and clean the signals.** We want to work on a specific electrode of EEG signal hereby mentioned as Cz electrode. One point to keep in mind is that P300 has frequency components up to the beta band of EEG (that ends at 35 Hz). We design a filtering routine to clean the signal. The same filtering should be applied to both training and testing EEG data when we create the classifier. In EEG pre-processing, often a spatial filtering routine is employed to further clean the signals. The simplest way to implement a spatial filter is to subtract from the channel of interest the average of the 4 closest channels (top, bottom, left and right).
2) **Segment and plot the training data.** Now we want to extract each single non-stimulus and stimulus epoch. A non-stimulus epoch starts each time the *StimulusSignal_training* transitions from 0 to 1 and lasts 650 ms (calculate the equivalent in samples). Similarly, a stimulus epoch starts each time the *StimulusSignal_training* transitions from 0 to 2 and lasts 650 ms. Then we Plot the average of the non-stimulus and stimulus trials
3) **Select and extract the features from the training data.** We look at the differences between the stimulus and non-stimulus trials and select a certain number of features to be used for classified. Once we have selected a number of features, we perform a feature ranking methodology (e.g. Fisher's score, or corrcoeff) to prune features that are less informative or keep just few features that are not highly correlated.
4) **Segment and extract the features from the testing data.** Now we want to segment the unlabelled epochs of the testing data and to calculate the features we have selected at step 3 on each of those epochs. Again, we need to remember an epoch starts each time the *FlashingSignal_testing* transitions from 0 to 1 and lasts 650 ms.
5) **Build and evaluate the classifier.** Now that we have built both the training and the testing set, we build a classifier.
   Note that we need to create a vector *train_stimulus* encoding information on which instances of the training data are relative to non-stimulus (group 1) or stimulus (group 2) trials. Once the classifier is made, we evaluate its performance in terms of accuracy, specificity and sensitivity by comparing the output of the classifier and the *solution* vector.

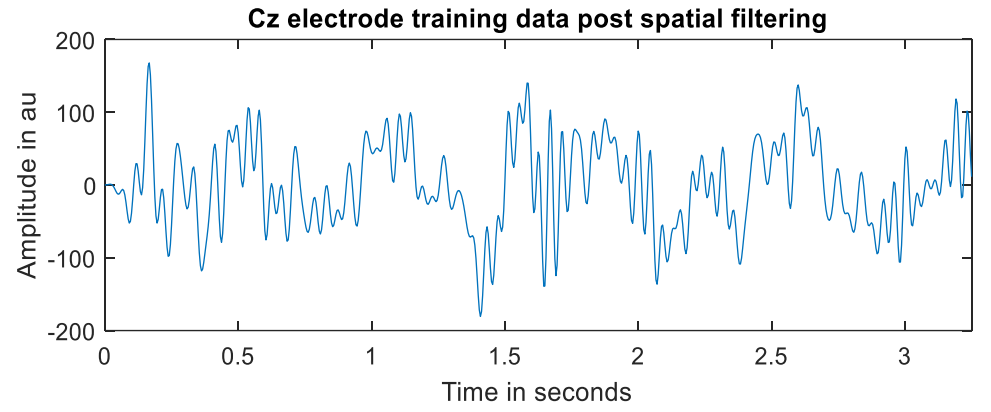
Outputs:

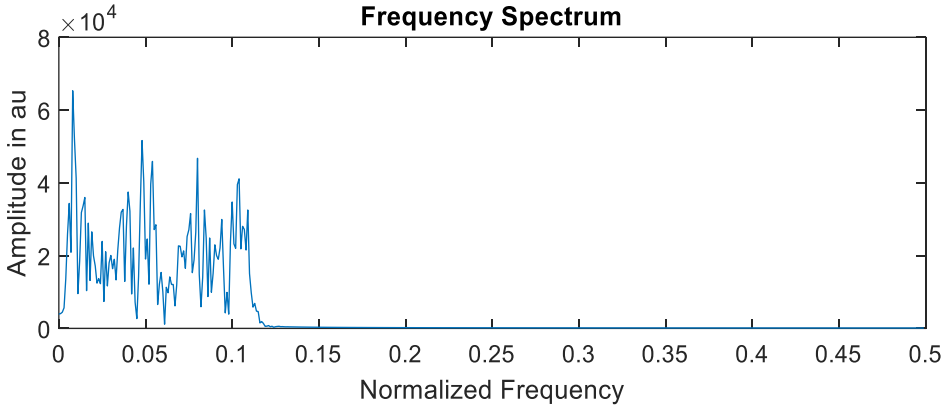

Raw EEG training data of Cz Electrode
Amplitude in au
Time in seconds

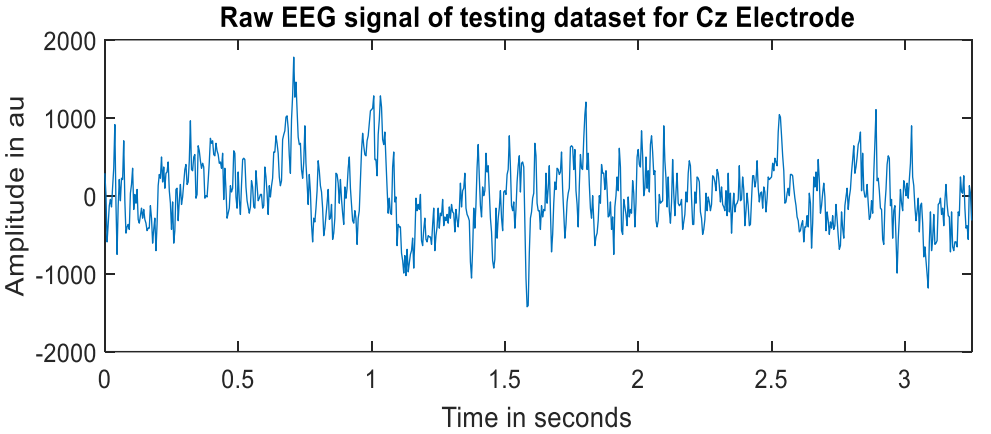

Frequency Spectrum
×10^4
Amplitude in au
Normalized Frequency

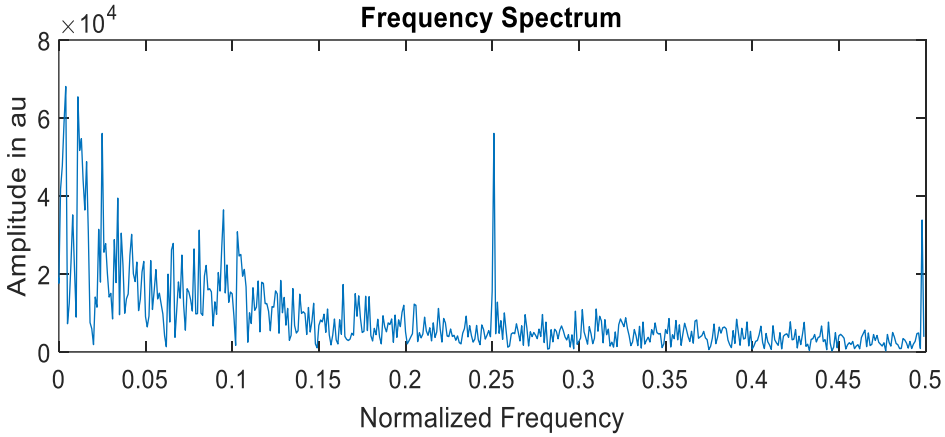

Bandpass filtered EEG training data of Cz Electrode
Amplitude in au
Time in seconds

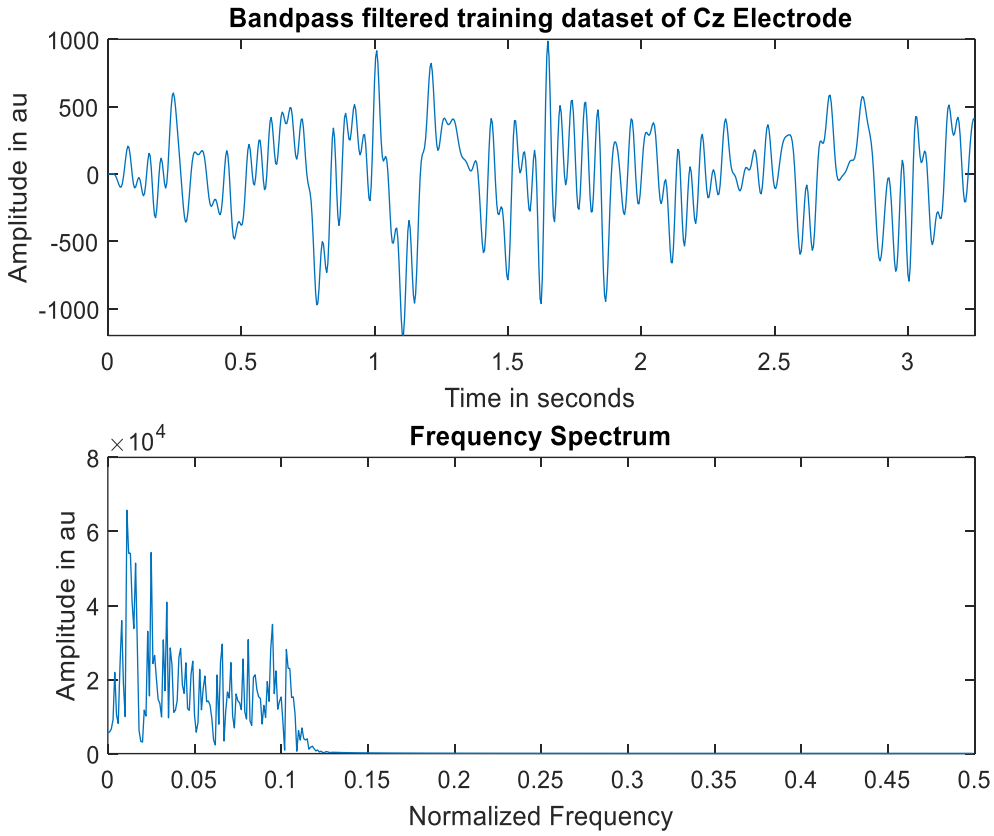

Frequency Spectrum
×10^4
Amplitude in au
Normalized Frequency

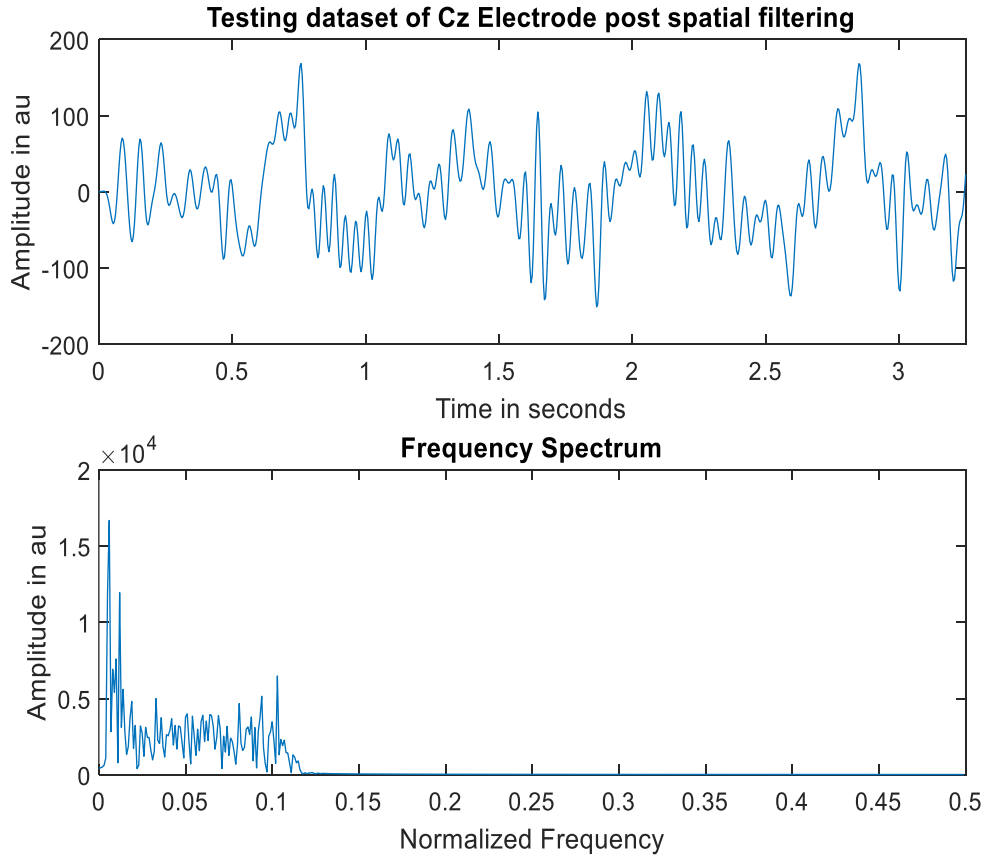

Cz electrode training data post spatial filtering
Amplitude in au
200
100
0
-100
-200
0
0.5
1
1.5
2
2.5
3
Time in seconds

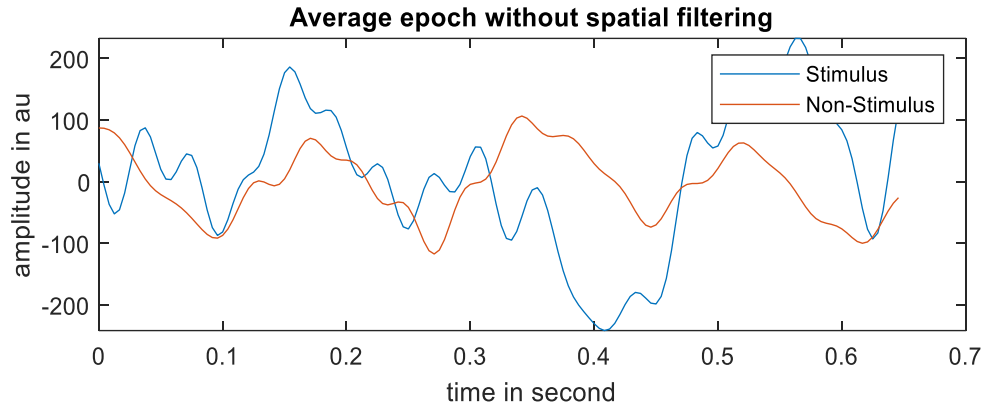

Frequency Spectrum
×10^4
Amplitude in au
8
6
4
2
0
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Normalized Frequency

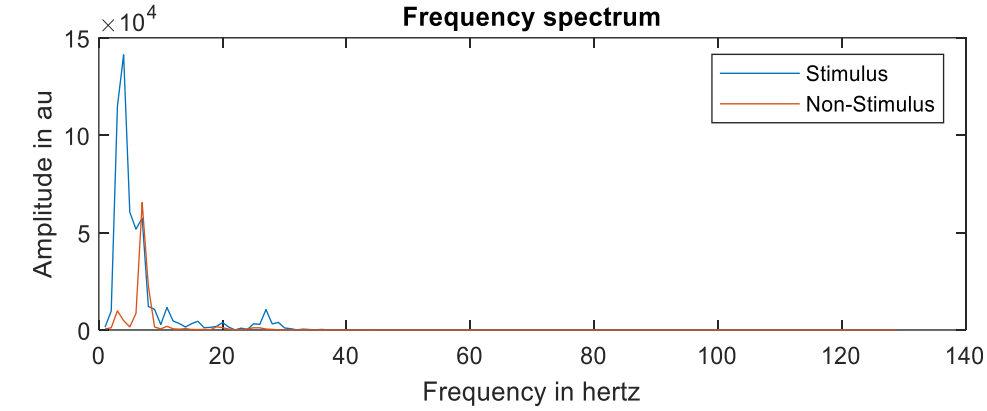

Raw EEG signal of testing dataset for Cz Electrode
Amplitude in au
2000
1000
0
-1000
-2000
0
0.5
1
1.5
2
2.5
3
Time in seconds

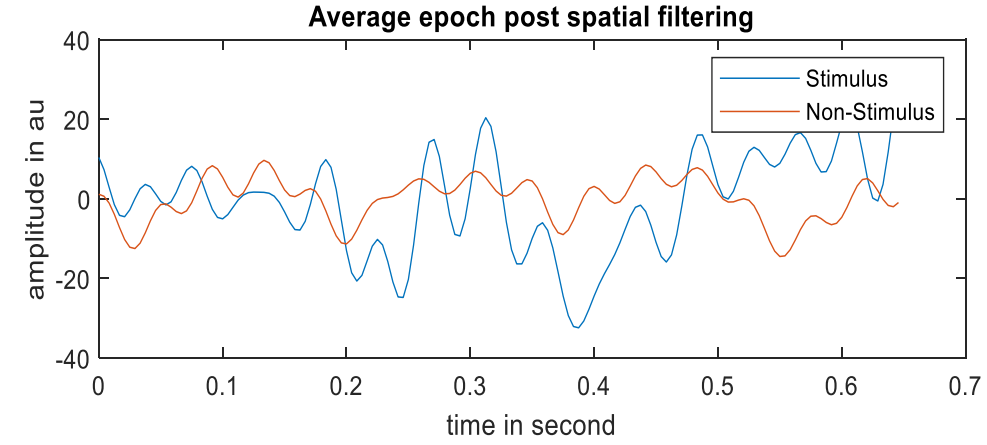

Frequency Spectrum
×10^4
Amplitude in au
8
6
4
2
0
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Normalized Frequency

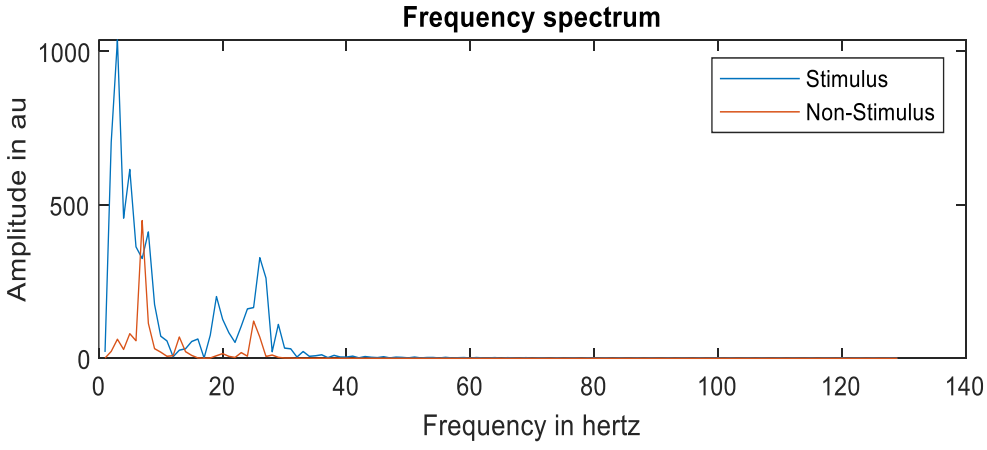

Bandpass filtered training dataset of Cz Electrode
Amplitude in au
1000
500
0
-500
-1000
0
0.5
1
1.5
2
2.5
3
Time in seconds
Frequency Spectrum
×10^4
8
6
4
2
0
Amplitude in au
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Normalized Frequency


Testing dataset of Cz Electrode post spatial filtering
Amplitude in au
200
100
0
-100
-200
0
0.5
1
1.5
2
2.5
3
Time in seconds
Frequency Spectrum
×10^4
2
1.5
1
0.5
0
Amplitude in au
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Normalized Frequency


Average epoch without spatial filtering
amplitude in au
time in second
Stimulus
Non-Stimulus


Frequency spectrum
×10^4
Amplitude in au
Frequency in hertz
Stimulus
Non-Stimulus


Average epoch post spatial filtering
amplitude in au
time in second
Stimulus
Non-Stimulus


Frequency spectrum
Amplitude in au
Frequency in hertz
Stimulus
Non-Stimulus

- Enter the classifier number (1-3) whose result you would like to view1

  The details of classifier are:

  Accuracy: 71.3889%

  Sensitivity: 26.6667%

  Specificity: 80.3333%

- Enter the classifier number (1-3) whose result you would like to view2

  The details of classifier are:

  Accuracy: 67.5%

  Sensitivity: 30%

  Specificity: 75%

- Enter the classifier number (1-3) whose result you would like to view3

  The details of classifier are:

  Accuracy: 60.8333%

  Sensitivity: 36.6667%

  Specificity: 65.6667%

Table:

Classifier No	Accuracy	Sensitivity	Specificity
1	71.389%	26.67%	80.33%
2	67.5%	30%	75%
3	60.83%	36.67%	65.67%

Explanation:

- We start with reading the .mat file.
- Each of the vectors present inside the structure are extracted into separate vectors for ease of calculation
- Upon observing the frequency spectrum of the Cz Electrode for training as well as the testing dataset, we observe that it has humongous amounts of noise
- To get rid of this noise, we use a band pass filter with details:

```
Fstop1 = 0.01;         % First Stopband Frequency
Fpass1 = 1;          % First Passband Frequency
Fpass2 = 25;           % Second Passband Frequency
Fstop2 = 32.5;    %35      % Second Stopband Frequency
```

- These specific values have been identified by several hundred hit and trial to get the best results from the classifier generated later on, Initially our intention was to keep it in the range of 1-30 hertz

- Afterwards, there was still a lot of noise observed in both the dataset, which could be credited to the electrode picking up the signal of nearby region as well, so to get rid of that, we apply a spatial filter taking in account adjacent 4 electrodes which gives us very good results in both training and testing dataset with some of the epochs even observed regularly.
- The display of the signals was limited to a few seconds so that data could be better visualized although clicking and dragging it would allow the user to see the data further.
- Afterwards, we focused on the marker vectors i.e. flashing signal testing and stimulus signal training to identify the transition points in them and then extract the respective epochs into separate vectors:
  sepoch-stimulus epoch (30 counts)
  nsepoch-nonstimulus epoch 150 counts)
  tepoch (unknown testing epoch 360counts)
  epoch-combined (150 counts of non-stimulus epoch for training followed by 30 counts of stimulus epoch)
  fepoch-epoch converted into frequency domain
  z1-identifier for stimulus epoch (30 counts of '1')
  z2-identifier for non-stimulus epoch (150 counts of '0')
  z0-identifier for epoch (z2 followed by z0)
- Earlier while extraction of data, we had subtracted 1 from solution vector so that even that vector denotes 0 for non-stimulus and 1 for stimulus epochs
- Next, we average all the stimulus and non-stimulus epochs post and pre spatial filtering, separately, where we observe that spatial filtering increases the divide between stimulus and non-stimulus epochs
- Afterwards, we use chi square tests in order to rank the features of epochs post spatial filtering using the command `fscchi2 for feature selection`
- This is done in both, time domain as well as frequency domain, so that classifiers can be trained in both domains and then the best classifiers will be saved and the rest will be discarded.
- Then we compare different kinds of classifiers using the classification learner app for different sets of features coming from frequency domain as well as time domain.
- Unfortunately, even after different kinds of iteration and changes, as per suggestion, we couldn't generate a classifier with exact details as required. But, we were able to generate three different classifiers close to that range, among which Classifier 2 shows most promising results. More stimulus epochs are needed to train a better classifier as the main problem is that in our training dataset 83.3% epochs belong to non-stimulus epochs so we lack enough stimulus epochs for classifiers to distinctly identify them.
- Afterwards, we use the solution vector and pass our testing epochs in the prediction function of the trained classifiers and validate the results.