a

“Course: Digital Signal Processing”

“Project: Cleaning EEG data using filters.”

Submitted To: Sir Shehzad Amin Sheikh

Sir Syed Moiz

Submitted By:

MUHAMMAD AHTISHAM SUDHEER 372982

MUHAMMAD ASIM KHAN 387724

SALEEM SHAHZAD 398364

LAIBA JABBAR 373314

DE-43 Syndicate-B

Submission Date:

3rd June ,2024

DEPARTMENT OF ELECTRICAL ENGINEERING

CEME, National University of Science and Technology (NUST)

Project Title:

"Cleaning EEG data using Filters."

Abstract:

Electroencephalography (EEG) is a critical diagnostic tool that captures electrical activity in the brain via electrodes placed on the scalp. This project focuses on cleaning EEG data from various forms of noise, especially line noise, using Digital Signal Processing (DSP) techniques in MATLAB. We implemented and evaluated several filters, including notch, high pass, low pass, and band pass filters, to isolate and preserve critical brainwave frequencies: delta, theta, alpha, sigma, and beta. Through comprehensive filtering and analysis, we unveil clearer, more interpretable EEG data, enabling a deeper understanding of brain function. This report documents the methodology, implementation, and results, demonstrating the effectiveness of the applied filtering techniques in noise reduction and signal analysis.

1.0. Introduction

1.1. Background and Motivation

Electroencephalography (EEG) is a vital tool in neuroscience and clinical diagnostics, offering a non-invasive means to monitor brain activity. Despite its significance, EEG data are often contaminated with various forms of noise, including line noise from power sources, which can obscure meaningful brain signals. Effective noise reduction is crucial for accurate analysis and interpretation. This project is motivated by the need to enhance EEG signal quality, facilitating better diagnosis, research, and understanding of brain function.

1.2. Objectives

1. To install and configure MATLAB and EEG Lab for EEG data processing.
2. To prepare and conduct preliminary analysis of EEG data, including detection and removal of line noise.
3. To understand and apply Power Spectral Density (PSD) analysis to evaluate the frequency content of EEG signals.
4. To design and implement various digital filters, including IIR and FIR filters, tailored for EEG signal processing.
5. To analyse EEG data across different brainwave frequency bands: delta, theta, alpha, sigma, and beta, using the designed filters.
6. 5. To select and analyse EEG data from 10 specific channels.
7. To compile a comprehensive report documenting the methodology, implementation, results, and analysis.

1.3. Scope of the Project

The scope of this project encompasses the end-to-end process of EEG data cleaning using digital signal processing techniques in MATLAB. This includes the initial setup, data preparation, Preliminary analysis, design and application of filters, and detailed analysis of filtered signals across key frequency bands. While the primary focus is on improving EEG signal quality for neurological research and clinical diagnostics, the techniques and methodologies developed are broadly applicable to other fields involving signal processing. These include biomedical engineering, cognitive neuroscience, sleep research, and even brain-computer interface development. The project's outcomes aim to provide a robust framework for future research and practical applications in these and related areas.

2.0. Literature Review

2.1. Overview of EEG

Electroencephalography (EEG) is a widely used technique for recording electrical activity of the brain. According to Niedermeyer and da Silva, EEG involves placing electrodes on the scalp to detect voltage fluctuations resulting from ionic current flows within the neurons of the brain **[1].** EEG is instrumental in diagnosing neurological conditions such as epilepsy, sleep disorders, and brain injuries, as well as in research on brain function and cognitive processes **[2].** EEG signals are characterized by different frequency bands, each associated with specific brain states and activities, such as delta waves in deep sleep and beta waves in active thinking **[3].**

2.2. Common Noise in EEG Data

EEG data are often contaminated by various types of noise that can obscure the true brain signals. As noted by Urigüen and Garcia-Zapirain, common sources of noise include power line interference (typically at 50 or 60 Hz), muscle artifacts (electromyographic noise), eye movements (electrooculographic noise), and environmental noise **[4].** These artifacts can significantly impact the quality of the recorded signals, making it challenging to accurately interpret the data. Croft and Barry emphasized that effective noise reduction techniques are essential to improve the signal-to-noise ratio and ensure reliable analysis of EEG signals **[5].**

2.3. Digital Signal Processing Techniques for EEG

Digital Signal Processing (DSP) techniques are crucial for analysing and enhancing EEG data. Key DSP methods include Fast Fourier Transform (FFT) for frequency analysis, Power Spectral Density (PSD) for understanding signal power distribution across frequencies, and various filtering techniques to isolate desired signal components while removing noise. According to Sanei and Chambers, FIR (Finite Impulse Response) and IIR (Infinite Impulse Response) filters are commonly used to design specific frequency filters, such as notch filters to eliminate power line noise and bandpass filters to focus on EEG frequency bands **[2].** These techniques are vital for improving signal quality and extracting relevant brainwave information.

2.4. Importance of Filtering in EEG Analysis

Filtering is a fundamental step in EEG signal processing, as it helps in extracting meaningful information from noisy data. Proper filtering can significantly enhance the quality of EEG signals by removing unwanted artifacts and preserving essential brainwave components. Widmann et al. highlighted that filters such as notch filters, high-pass, low-pass, and bandpass filters are designed to target specific noise sources and frequency bands, facilitating clearer and more reliable EEG analysis **[6].** This is particularly important for clinical diagnostics and research, where accurate interpretation of EEG data is critical.

3.0. Methodology

3.1. Installation of MATLAB

Install MATLAB if hasn't been installed by visiting math.
https://www.mathworks.com/help/install/ug/install-products-with-internet-connection.html
In our case it was already installed.

3.1.2. Installation Steps for EEG Lab

Installing the EEG Lab toolbox involves the following steps:

1. Download: Visit the official EEG Lab website and navigate to the download section. Select the appropriate version of the toolbox compatible with our MATLAB version and operating system. https://sccn.ucsd.edu/eeglab/downloadtoolbox.php/

2. Extract Files: After downloading the EEG Lab toolbox archive, extract its contents to a designated folder on our computer.

3. Add Path: Open MATLAB and navigate to the "Set Path" option in the "File" menu. Add the path to the extracted EEG Lab toolbox folder to MATLAB's search path to enable access to its functions and scripts.

4. Verify Installation: To verify the successful installation of the EEG Lab toolbox, type "eeglab" in the MATLAB command window. If installed correctly, EEG Lab's graphical user interface (GUI) should open without any errors.

By following these steps, MATLAB can be effectively configured with the EEG Lab toolbox, providing a comprehensive platform for EEG data processing and analysis.

3.2. Data Preparation and Preliminary Analysis

3.2.1. Description of the EEG Dataset

The EEG dataset utilized in this project consists of multichannel EEG recordings acquired from scalp electrodes. Key attributes of the dataset include the number of channels (nbchannels), the EEG data itself, and the sampling rate (srate), which determines the temporal resolution of the recordings. The EEG data is organized as a matrix, where rows represent individual channels, and columns represent time samples.

3.2.2. Detection of Line Noise

Detection and removal of line noise, typically occurring at power line frequencies (e.g., 50 or 60 Hz), are crucial for ensuring the quality of EEG data. This is accomplished through Power Spectral Density (PSD) analysis, which provides insight into the frequency content of the signal. We start by defining a time window for analysis, converting it into sample indices, and calculating the number of samples in the window. Then, we generate a frequency vector for the Fast Fourier Transform (FFT).

The **Power Spectral Density (PSD)** is computed for each channel within the specified time window using the FFT. PSD represents the distribution of power across different frequencies in the signal. In our implementation, we compute the single-sided spectrum from the two-sided spectrum obtained from the FFT.

Next, the average PSD across all channels is calculated. Peaks in the PSD corresponding to line noise frequencies, such as 50 Hz and 60 Hz, are identified. These peaks indicate the presence of line noise in the EEG data. By estimating the PSD, peaks corresponding to line noise frequencies can be identified and attenuated.

In our implementation, we computed the PSD using the **Welch method**, a popular technique for estimating PSD from finite-length time series data. This was achieved by segmenting the EEG data into overlapping windows, applying a windowing function, and computing the PSD using FFT. Peaks in the PSD corresponding to line noise frequencies were then targeted for removal using notch filters.

3.3. Filter Design and Implementation

3.4.IIR filters:

IIR Filters are a class of digital filters characterized by feedback, allowing them to have an infinite impulse response. Compared to FIR filters, IIR filters typically require fewer coefficients and are computationally more efficient, making them suitable for real-time applications.

3.4.1. Notch Filter Design

Notch filters were designed to specifically target, and attenuate line noise frequencies identified in the PSD analysis. In our MATLAB code, IIR (Infinite Impulse Response) notch filters were implemented using the `**iirnotch**` function. This function allows for the design of notch filters with user-defined notch frequencies and quality factors, enabling precise suppression of line noise while preserving the EEG signal of interest.

3.5. Design of FIR Filters:

The design of FIR filters involves specifying the filter order (N) and the cutoff frequencies (Fc_delta). Here's a breakdown:

- **Filter Order (N):** The filter order determines the number of coefficients in the filter, which in turn affects the filter's frequency response and performance. A higher filter order generally results in sharper frequency cutoffs but requires more computational resources.

- **Cutoff Frequencies (Fc_delta):** For the LPF, Fc_delta(2) represents the upper cutoff frequency, while for the HPF, Fc_delta(1) represents the lower cutoff frequency. Together, these define the passband and stopband frequencies for each filter.

3.5.1.FIR Filter Design Function:

The **" fir1"** function is used to design FIR filters in MATLAB. Its parameters include the filter order (N), the normalized cutoff frequency (expressed as a fraction of the Nyquist frequency), the type of filter (e.g., 'low' for LPF, 'high' for HPF), and the windowing function (in this case, Hanning window).

3.5.2. Hanning Window:

The Hanning window is a type of windowing function commonly used in signal processing to shape the frequency response of FIR filters. It tapers the ends of the filter coefficients to reduce spectral leakage and improve the filter's frequency response. The Hanning window smooths the transition between the passband and stopband of the filter, enhancing its performance.

3.5.3. Filter Coefficients (a and b):

In FIR filters, the coefficients (b for the numerator coefficients and a for the denominator coefficients) represent the filter's impulse response.

3.5.5. High Pass Filter Design

High pass filters were designed to remove low-frequency components and baseline drift from the EEG signals below the cutoff while preserving higher frequencies associated with brainwave activity. **Baseline drifts** are gradual shifts in the EEG signal baseline over time, often caused by electrode impedance changes or movement artifacts, affecting data interpretation. FIR (Finite Impulse Response) high pass filters were implemented using the `**fir1**` function.

3.5.6. Low Pass Filter Design

Low pass filters were employed to suppress high-frequency noise and artifacts from the EEG signals. **Artifacts** in EEG data are irregularities caused by external sources like muscle activity or equipment malfunctions, distorting the signal. Like high pass filters, FIR low pass filters were designed using the `**fir1**` function, with parameters including the cutoff frequency and filter order. Low pass filters effectively attenuate frequencies above the cutoff while preserving lower frequency components relevant to EEG analysis.

3.5.7. Band Pass Filter Design

Band pass filters were designed to isolate specific frequency bands associated with different brainwave activities, such as delta, theta, alpha, sigma, and beta waves. These filters were implemented as combinations of high pass, low pass, and band pass filters, each targeting a specific frequency range. The Hanning window was utilized to design FIR filters, which provides improved frequency response and reduced spectral leakage compared to other windowing functions. Arrays of filter coefficients (a, b) were computed using the `**fir1**` function, specifying the desired frequency band and filter order.

3.6. Selection and Extraction of EEG Channels

Before applying filters, a subset of EEG channels was selected for analysis. This involved choosing channels with the most relevant information for the intended analysis, such as those covering regions of interest or exhibiting strong neural activity. Selected channels were extracted from the EEG data matrix for further processing.

3.7. Application of Filters on Selected Channels

Once the EEG channels were selected, the designed filters were applied to each channel to remove noise and isolate desired frequency components. This was achieved by convolving the EEG signals with the filter coefficients using MATLAB's `**filtfilt**` function, ensuring zero-phase distortion and preserving the temporal characteristics of the signals. After filtering, the cleaned EEG signals were ready for further analysis and interpretation.

4.0. Implementation:

All the above-mentioned steps were implemented using the MATLAB code given below.

4.1. MATLAB Code:

```
% Define the path to EEGLAB and add it to MATLAB's search path
eeglabPath = 'C:\Users\ahtis\OneDrive - National University of Sciences &
Technology\Semester\SEMESTER 6\DSP\LAB\project0\eeglab2024.0';
addpath(eeglabPath);

% Load the EEG dataset
eegDataset = '825_2_PD_REST.mat';
load(eegDataset);
%%
% Extract the sampling frequency (in Hz)
Fs = EEG.srate;

% Set the analysis time window in seconds
startTime = 0;
endTime = 10;

% Convert the time window to sample indices
```

```
startIndex = round((startTime - EEG.xmin) *Fs) + 1;
endIndex = round((endTime - EEG.xmin) * Fs);

% Determine the number of samples in the analysis window
sampleCount = endIndex - startIndex + 1;

% Generate the frequency vector for the FFT
freqVector = (0:(sampleCount/2)) * (Fs / sampleCount);

% Prepare a matrix to hold the PSD values for each channel
channelPSDs = zeros(length(freqVector), EEG.nbchan);

% Calculate the PSD for each channel within the specified time window
for channelIndex = 1:EEG.nbchan
    % Extract the data for the current channel and time window
    currentData = EEG.data(channelIndex, startIndex:endIndex);

    % Perform the Fast Fourier Transform (FFT) on the data
    fftResult = fft(currentData);

    % Compute the two-sided spectrum and then the single-sided spectrum
    twoSidedSpectrum = abs(fftResult / sampleCount);
    singleSidedSpectrum = twoSidedSpectrum(1:(sampleCount/2 + 1));
    singleSidedSpectrum(2:end-1) = 2 * singleSidedSpectrum(2:end-1);

    % Store the PSD for the current channel
    channelPSDs(:, channelIndex) = singleSidedSpectrum;
end

% Calculate the average PSD across all channels
averagePSD = mean(channelPSDs, 2);

% Find the PSD values at 50 Hz and 60 Hz
psdAt50Hz = averagePSD(round(50 / (Fs / sampleCount)) + 1);
psdAt60Hz = averagePSD(round(60 / (Fs / sampleCount)) + 1);

% Output the PSD values at 50 Hz and 60 Hz
fprintf('Average PSD at 50 Hz: %f\n', psdAt50Hz);
fprintf('Average PSD at 60 Hz: %f\n', psdAt60Hz);

% Create a plot of the average PSD
figure;
plot(freqVector, 10 * log10(averagePSD));
title('Average Power Spectral Density (PSD) Across All Channels');
xlabel('Frequency (Hz)');
ylabel('Power/Frequency (dB/Hz)');
axis([0, Fs/2, -100, 50]);
grid on;
%%
% Define notch filter parameters for 60 Hz and its harmonics
notchFreqs = 60:60:(Fs/2); % Frequencies to remove
Q = 35; % Quality factor

% Design and apply notch filters for each harmonic
for freq = notchFreqs
    [b, a] = iirnotch(freq/(Fs/2), freq/(Fs/2)/Q);
    EEG.data = filtfilt(b, a, EEG.data')'; % Apply the filter to the data
end
```

```
%%
% Compute the PSD after filtering
[psd, f] = pwelch(EEG.data', hamming(EEG.pnts), [], [], Fs);

mean_psd = mean(psd, 2);
% Plot the PSD after notch filtering
figure;
plot(f, 10*log10(mean_psd));
title('Average Power Spectral Density (PSD) Across All Channels After Notch
Filtering');
xlabel('Frequency (Hz)');
ylabel('Power/Frequency (dB/Hz)');
axis([0, Fs/2, -100, 200]);
grid on;

% Extract the updated EEG data after notch filtering
updatedEEGData = EEG.data;

% List available channels (1 to 67)
nbchan = 1:67;

% Select 10 channels from the updated EEG data
selectedChannels = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
eegSelected = updatedEEGData(selectedChannels, :);
disp('Selected Channels:');
disp(selectedChannels);

%%
%Plot the combined EEG data for selected channels
figure;
hold on;
colors = lines(10); % Get distinct colors for each plot
for i = 1:10
    plot(eegSelected(i, :), 'Color', colors(i, :));
end
hold off;
title('Combined EEG Data for Selected 10 Channels After Notch Filter');
xlabel('Time (samples)');
ylabel('Amplitude');
legend(arrayfun(@(x) ['Channel ' num2str(selectedChannels(x))], 1:10,
'UniformOutput', false));
grid on;

%%
% Generate the frequency vector for the FFT
freqVector = (0:(sampleCount/2)) * (Fs / sampleCount);

% Define filter orders
N = 1600; % Filter order

% Define the cut-off frequencies for each filter (as a fraction of the Nyquist
frequency)
Fc_delta = [0.5 4] / (Fs/2); % Delta band
Fc_theta = [4 7] / (Fs/2);   % Theta band
Fc_alpha = [8 12] / (Fs/2);   % Alpha band
Fc_sigma = [12 16] / (Fs/2);  % Sigma band
Fc_beta = [13 30] / (Fs/2);   % Beta band

% Initialize filtered EEG data matrix (3D matrix)
```

```
filteredEEGData = zeros(EEG.nbchan, sampleCount, 5);
% Filter each selected EEG channel for each frequency band
for i = 1:length(selectedChannels)
    channelIndex = selectedChannels(i);
    currentData = EEG.data(channelIndex, startIndex:endIndex);

    % Delta band filter (combination of low pass and high pass filters)
    b_LPF_delta = fir1(N, Fc_delta(2), 'low', hanning(N+1)); % Low pass filter
    b_HPF_delta = fir1(N, Fc_delta(1), 'high', hanning(N+1)); % High pass filter
    filtered_delta = filtfilt(b_LPF_delta, 1, filtfilt(b_HPF_delta, 1,
currentData));

    % Theta band filter (band pass filter)
    b_BPF_theta = fir1(N, Fc_theta, 'bandpass', hanning(N+1)); % Band pass filter
    filtered_theta = filtfilt(b_BPF_theta, 1, currentData);

    % Alpha band filter (combination of low pass, high pass, and band pass
filters)
    b_LPF_alpha = fir1(N, Fc_alpha(2), 'low', hanning(N+1)); % Low pass filter
    b_HPF_alpha = fir1(N, Fc_alpha(1), 'high', hanning(N+1)); % High pass filter
    b_BPF_alpha = fir1(N, Fc_alpha, 'bandpass', hanning(N+1)); % Band pass filter
    filtered_alpha = filtfilt(b_LPF_alpha, 1, filtfilt(b_HPF_alpha, 1,
filtfilt(b_BPF_alpha, 1, currentData)));

    % Sigma band filter (band pass filter)
    b_BPF_sigma = fir1(N, Fc_sigma, 'bandpass', hanning(N+1)); % Band pass filter
    filtered_sigma = filtfilt(b_BPF_sigma, 1, currentData);

    % Beta band filter (combination of low pass, high pass, and band pass filters)
    b_LPF_beta = fir1(N, Fc_beta(2), 'low', hanning(N+1)); % Low pass filter
    b_HPF_beta = fir1(N, Fc_beta(1), 'high', hanning(N+1)); % High pass filter
    b_BPF_beta = fir1(N, Fc_beta, 'bandpass', hanning(N+1)); % Band pass filter
    filtered_beta = filtfilt(b_LPF_beta, 1, filtfilt(b_HPF_beta, 1,
filtfilt(b_BPF_beta, 1, currentData)));

    % Store the filtered data for each frequency band in the 3D matrix
    filteredEEGData(i,:,:) = [filtered_delta; filtered_theta; filtered_alpha;
filtered_sigma; filtered_beta]';
end
%%

% Define the titles for each frequency band
bandTitles = {'Delta', 'Theta', 'Alpha', 'Sigma', 'Beta'};
bandColors = {'b', 'r', 'g', 'm', 'c'}; % Colors for each band

% Channels to plot results for
channelsToPlot = [1]; % Specify which channels to plot

for channelToPlot = channelsToPlot
    % Plot the filtered EEG data for the specified channels in the time domain
    figure;
    % Loop through each frequency band
    for bandIndex = 1:5
        subplot(5, 1, bandIndex);
        plot(filteredEEGData(channelToPlot, :, bandIndex), bandColors{bandIndex});
        title([bandTitles{bandIndex} ' Filtered EEG Data for Channel '
num2str(channelToPlot)]);
        xlabel('Time (samples)');
        ylabel('Amplitude');
```

```
        grid on;
    end
%%
    % Define the band names, colors, and cutoff frequencies for plotting
bands = {'Delta', 'Theta', 'Alpha', 'Sigma', 'Beta'};
colors = {'b', 'r', 'g', 'm', 'c'};
cutoffs = {[0.5, 4], [4, 7], [8, 12], [12, 16], [13, 30]};
behavioralTraits = {
    'Delta (0.5 to 4 Hz): Associated with deep sleep and restorative sleep
stages.',
    'Theta (4 to 7 Hz): Related to light sleep, relaxation, creativity, and
meditative states.',
    'Alpha (8 to 12 Hz): Indicative of relaxed wakefulness, reduced stress, and
enhanced learning.',
    'Sigma (12 to 16 Hz): Linked to sleep spindles during sleep, involved in
memory consolidation.',
    'Beta (13 to 30 Hz): Associated with active thinking, concentration, and
problem-solving.'
};

% Plot the filtered EEG data for selected channels in the frequency domain
figure;
for ch = 1:length(selectedChannels)
    channelToPlot = selectedChannels(ch);
    % Loop through each band and plot
    for i = 1:length(bands)
        subplot(5, 1, i);
        % Compute FFT
        fftData = fft(filteredEEGData(channelToPlot, :, i));
        fftData = abs(fftData / sampleCount);
        fftData = fftData(1:sampleCount/2+1);
        fftData(2:end-1) = 2 * fftData(2:end-1);
        % Plot the frequency domain data
        plot(freqVector, 10*log10(fftData), colors{i});
        title([bands{i} ' Filtered EEG Data (Freq Domain) for Channel '
num2str(channelToPlot)]);
        xlabel('Frequency (Hz)');
        ylabel('Power/F (dB/Hz)');

        grid on;

        % Add markers at the cutoff frequencies
        hold on;
        plot([cutoffs{i}(1), cutoffs{i}(1)], ylim, '--k', 'LineWidth', 1); % Start
cutoff frequency
        plot([cutoffs{i}(2), cutoffs{i}(2)], ylim, '--k', 'LineWidth', 1); % End
cutoff frequency
        hold off;

        % Print behavioral traits associated with each band
        fprintf('\nBehavioral traits for %s band:\n%s\n', bands{i},
behavioralTraits{i});
    end
    pause; % Pause to view each channel's plots
end
end
```

5.0. Results

5.1. Preliminary Analysis Results

The preliminary analysis unveiled crucial insights into the EEG data. Initial examination revealed notable characteristics such as line noise present within the signals. Understanding these preliminary findings served as a foundation for subsequent processing steps, guiding the selection and implementation of appropriate filtering techniques.

```
Average PSD at 50 Hz: 4.975779
Average PSD at 60 Hz: 51.653806
```

Fig-1.0: Checking Average PSD at 50 & 60 Hz to detect Channel Noise.

5.2. PSD Before and After Notch Filtering

Comparing the Power Spectral Density (PSD) before and after notch filtering demonstrated the efficacy of the noise reduction process. Pre-filtering PSD showcased prominent peaks corresponding to line noise frequencies, indicating contamination within the EEG signals. Conversely, post-filtering PSD exhibited significant attenuation at these frequencies, signifying successful removal of line noise and its harmonics.

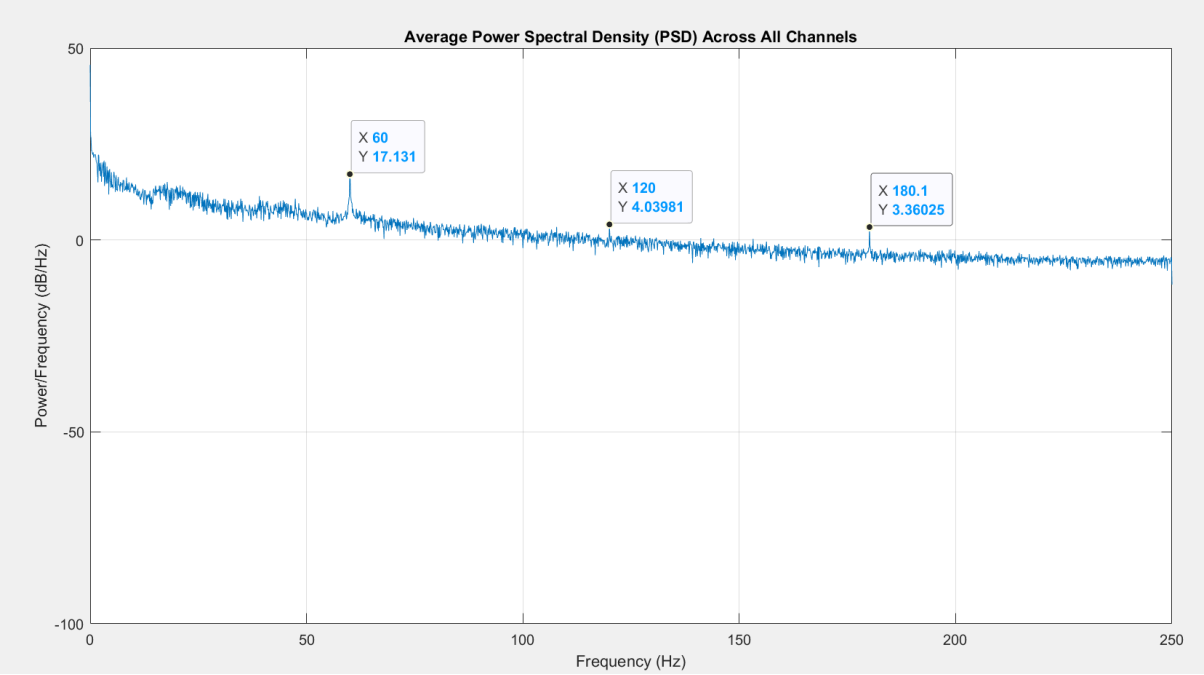

Fig-2.1: Average Power Spectral Density (PSD) across All channels showing peaks at 60 Hz and its harmonics

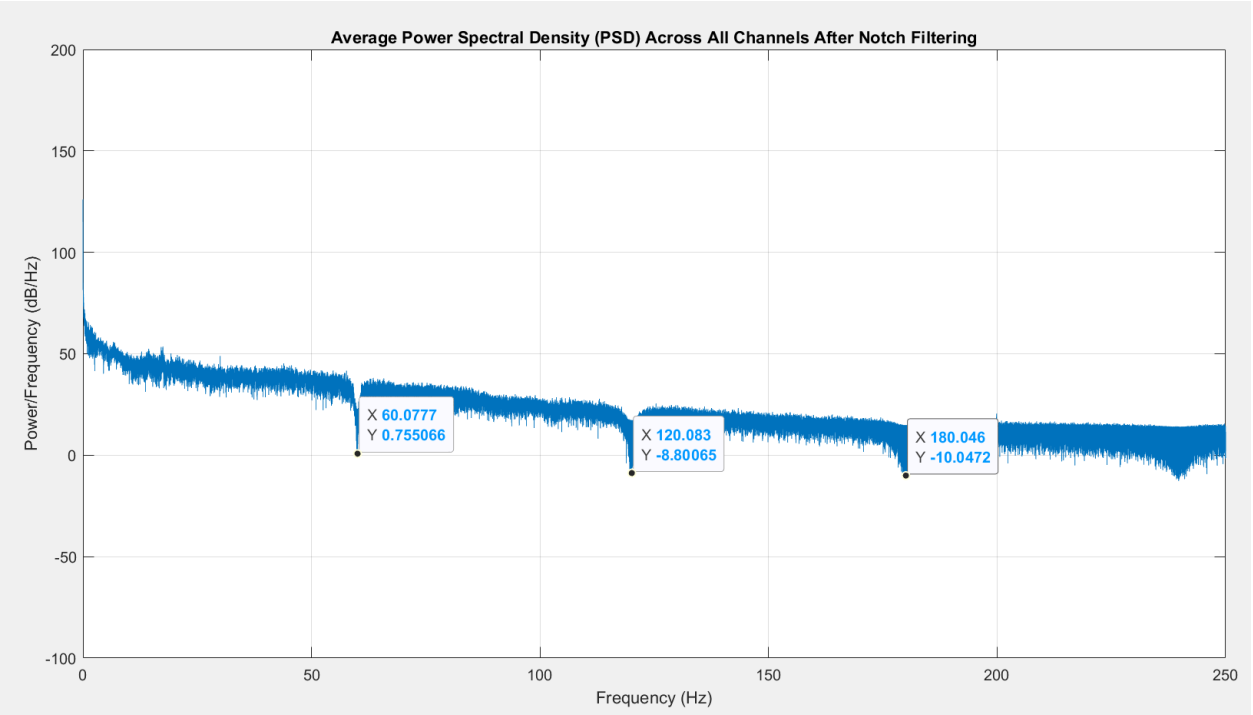

Fig-2.2.0: Average Power Spectral Density (PSD) across All channels after Notch filter creating dips at 60 Hz and its harmonics.

5.3. Selection of Channels for Further processing

10 channels were selected from channel no: 1- 10. We can select channels of our own choice by writing their number in defined array for channel selection.

```
Selected Channels:
     1     2     3     4     5     6     7     8     9    10
```

Fig-3.0: Selection of Channel 1-10 from available 67 Channels in data set

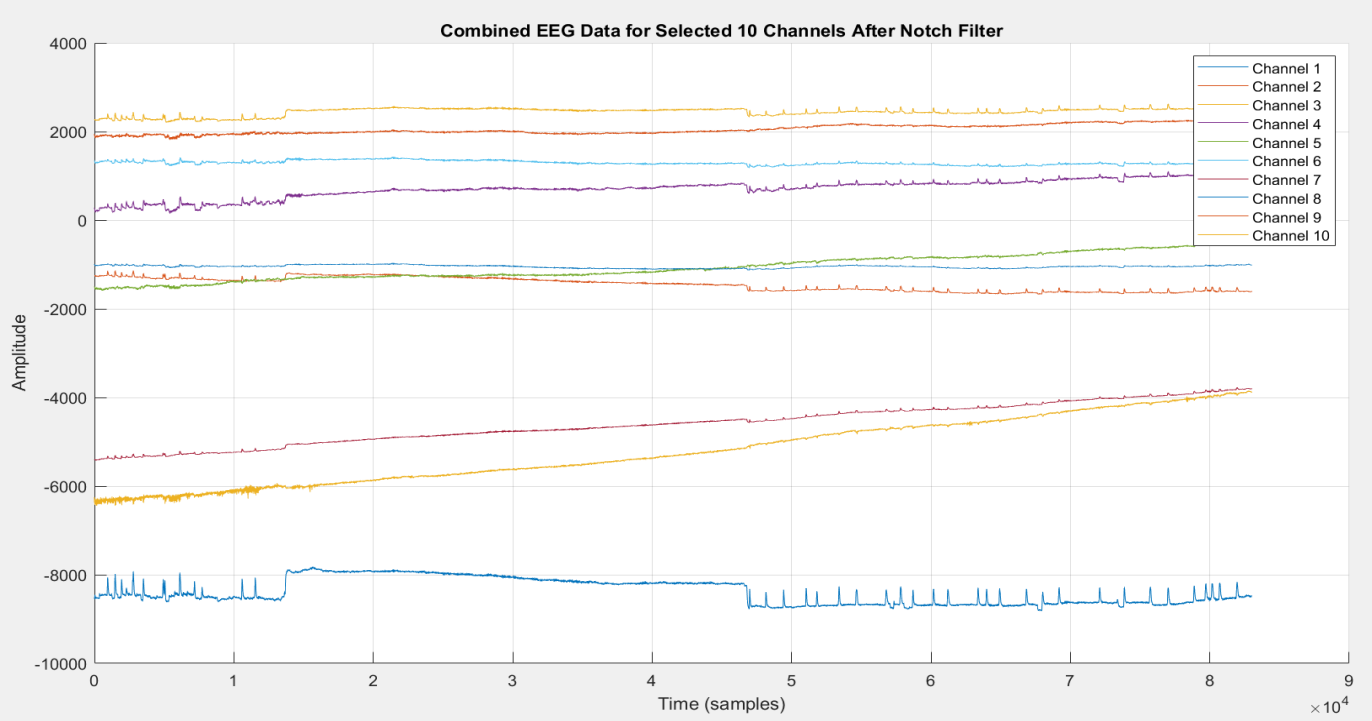

Fig-3.1: Combined EEG data for Selected 10 Channels

5.3. Filtered EEG Data for Selected Channels

Analysis of the filtered EEG data across selected channels provided valuable insights into the effectiveness of the applied filters. Visual inspection revealed the suppression of noise and artifacts, resulting in cleaner and more interpretable signals. This enhanced signal quality lays the groundwork for precise and reliable analysis of brainwave activities across different frequency bands.

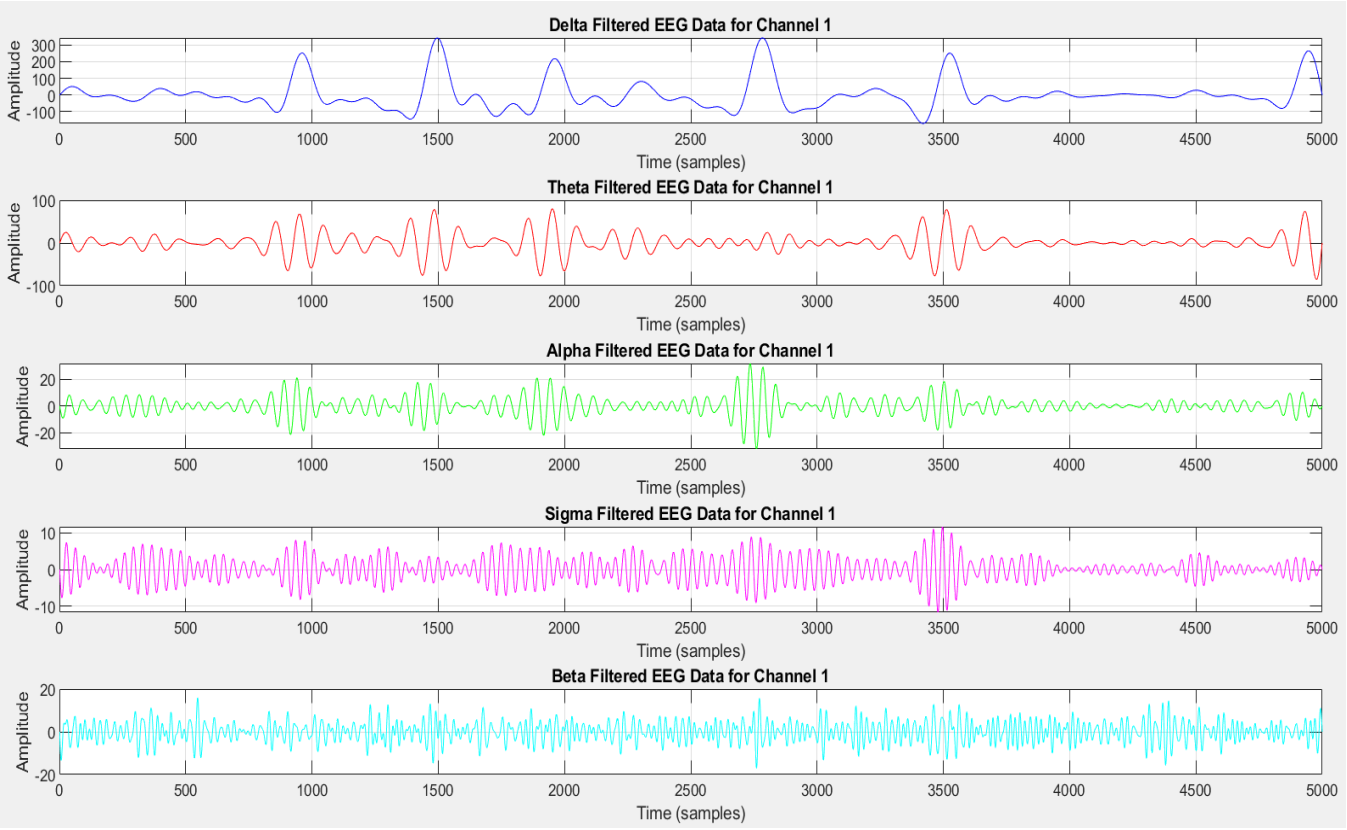

Fig-4.0: Filtered EEG Data after passing from Designed filters for Channel-1

5.4. Analysis of EEG Data in Different Frequency Bands

Further exploration involved analysing EEG data within distinct frequency bands, including delta, theta, alpha, sigma, and beta waves. Each frequency band exhibited characteristic patterns indicative of specific brain activities and cognitive states. By dissecting EEG signals into these frequency components, a comprehensive understanding of brain dynamics and functionality was attained.

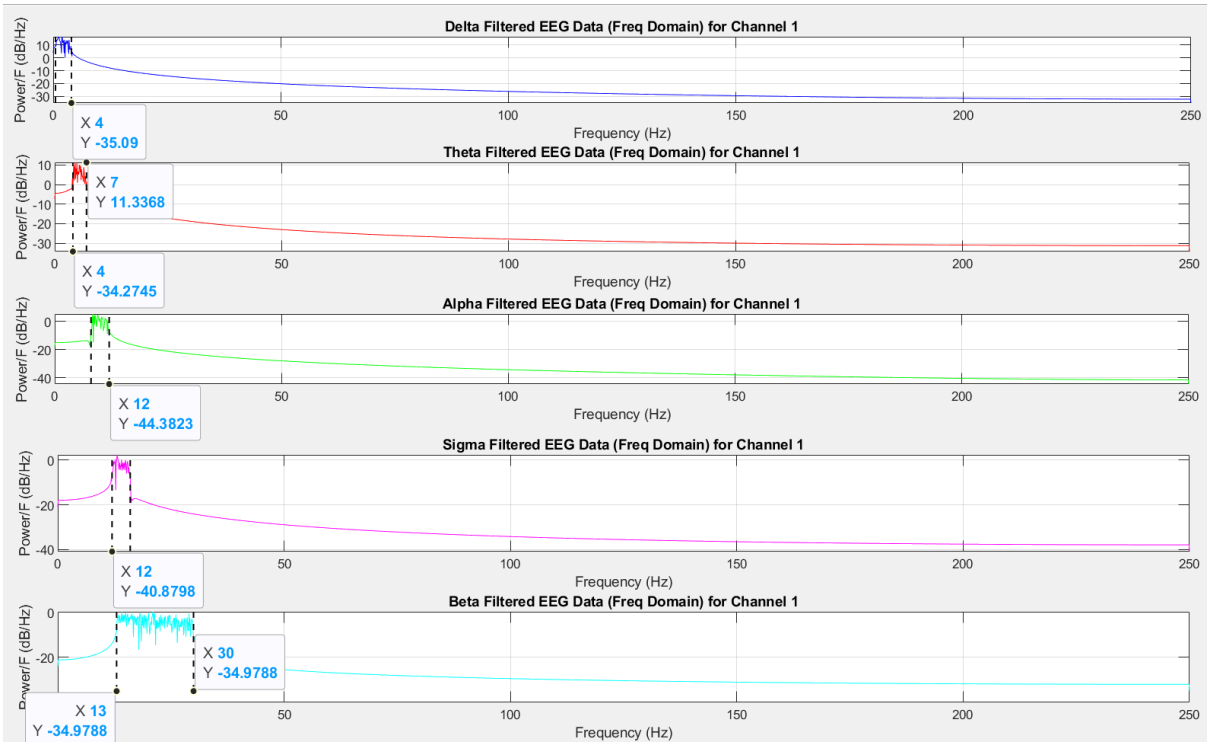

Fig-5.0: Freq response of EEG Data of channel-1 after passing from designed Freq Bands

5.5. Interpretation of Behavioural Traits from Filtered Data

Interpreting behavioural traits from the filtered EEG data provided valuable insights into the subject's cognitive and emotional states. Observations within different frequency bands revealed patterns consistent with known physiological responses, such as relaxation, concentration, and sleep stages. These interpretations elucidate the underlying neural mechanisms governing human behaviour and cognitive processes.

```
Behavioral traits for Delta band:
Delta (0.5 to 4 Hz): Associated with deep sleep and restorative sleep stages.

Behavioral traits for Theta band:
Theta (4 to 7 Hz): Related to light sleep, relaxation, creativity, and meditative states.

Behavioral traits for Alpha band:
Alpha (8 to 12 Hz): Indicative of relaxed wakefulness, reduced stress, and enhanced learning.

Behavioral traits for Sigma band:
Sigma (12 to 16 Hz): Linked to sleep spindles during sleep, involved in memory consolidation.

Behavioral traits for Beta band:
Beta (13 to 30 Hz): Associated with active thinking, concentration, and problem-solving.
```

Fig-6.1: Behavioural Traits for desired Bands

6.Conclusion

6.1. Summary of Findings

The comprehensive analysis of EEG data employing Digital Signal Processing techniques yielded significant findings. Through meticulous filtering, line noise and artifacts were effectively removed, enhancing the quality and interpretability of the signals. Examination across different frequency bands provided insights into brainwave activities, facilitating the understanding of cognitive and emotional states. Overall, the project achieved its objectives of cleaning EEG data and exploring brain activity patterns.

6.2. Future Work

To address the identified challenges and extend the scope of this research, several avenues for future work are proposed.

1. Refining filtering techniques to better adapt to individual EEG data characteristics and noise profiles could enhance the efficacy of noise reduction.
    - Adaptive Filtering.
    - Multi-Stage Filtering.
2. Exploring advanced machine learning algorithms for noise classification and removal could offer alternative approaches to signal processing.
    - Deep neural networks, including Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), have shown promise in noise classification and removal tasks.
    - Sparse coding algorithms, such as Sparse Autoencoders or Sparse Bayesian Learning, are adept at extracting meaningful features from EEG data while suppressing noise.
3. Investigating the integration of real-time feedback mechanisms could enable dynamic adjustment of filtering parameters, enhancing adaptability to changing EEG signal dynamics. Collaborative efforts with domain experts and interdisciplinary research endeavours could further enrich the understanding of EEG data processing and its applications in neuroscience and clinical diagnostics.

References:

1. Niedermeyer, E., & da Silva, F. L. (2004). Electroencephalography: Basic principles, clinical applications, and related fields. Lippincott Williams & Wilkins.
2. Sanei, S., & Chambers, J. A. (2007). EEG signal processing. John Wiley & Sons.
3. Niedermeyer, E. (1997). Alpha rhythms as physiological and abnormal phenomena. International Journal of Psychophysiology, 26(1-3), 31-49.
4. Urigüen, J. A., & Garcia-Zapirain, B. (2015). EEG artifact removal—state-of-the-art and guidelines. Journal of Neural Engineering, 12(3), 031001.
5. Croft, R. J., & Barry, R. J. (2000). Removal of ocular artifact from the EEG: a review. Neurophysiologie Clinique/Clinical Neurophysiology, 30(1), 5-19.
6. Widmann, A., Schröger, E., & Maess, B. (2015). Digital filter design for electrophysiological data–a practical approach. Journal of Neuroscience Methods, 250, 34-46.

Appendix:

A: Data Set Mat File
https://nustedupk0-my.sharepoint.com/:u:/g/personal/msudheer_ee43ceme_student_nust_edu_pk/ERXTvdmEO4JLgjFCfP_TXp8BWC6C-VJm2bUQczllOA2Uow?e=JrHT5M

B: MATLAB Complete code:
https://nustedupk0-my.sharepoint.com/:u:/g/personal/msudheer_ee43ceme_student_nust_edu_pk/ERoQnuQtMpFPmLqMgQutJLoBBCeaQ6sIEFatoWCdibWGsA?e=OG8veo

---