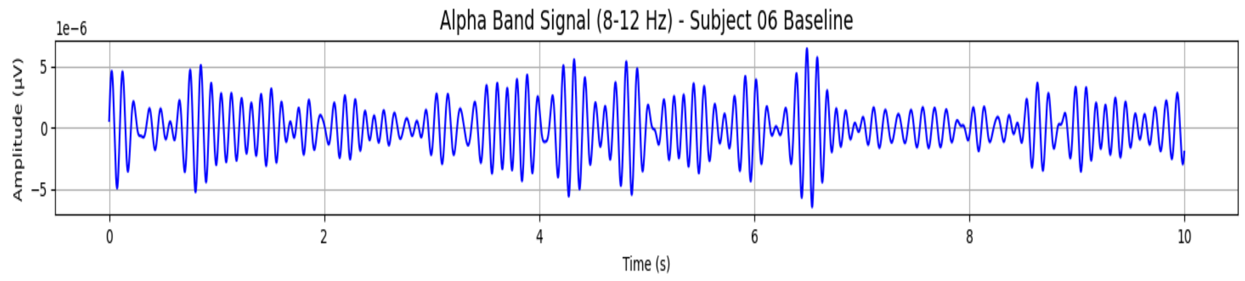
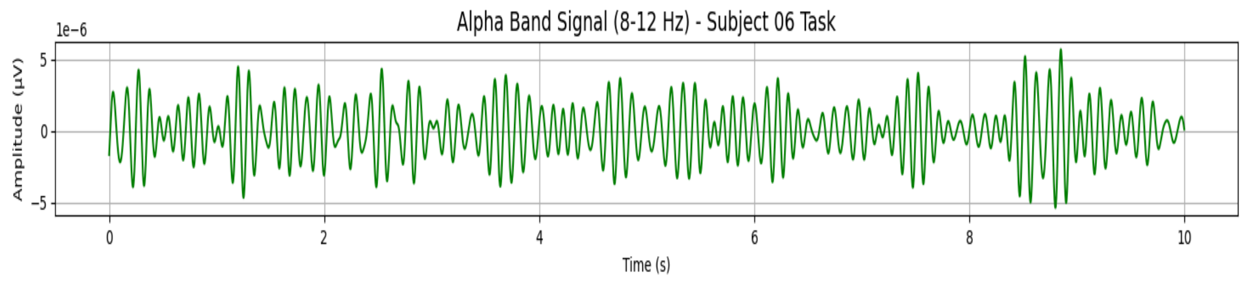
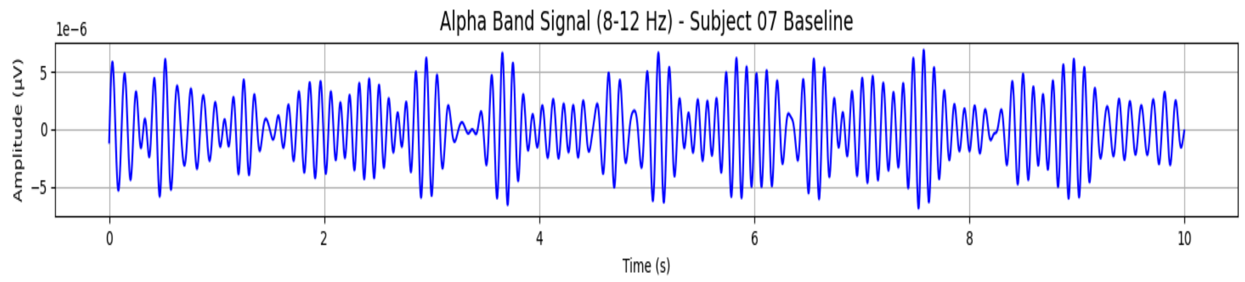
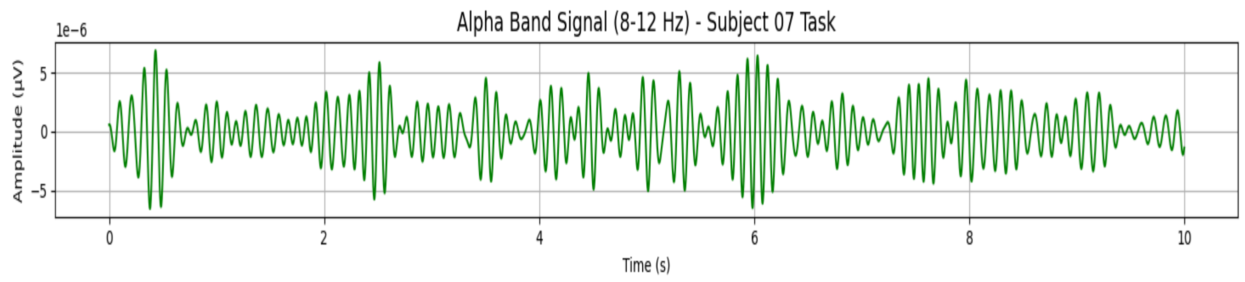
Question 3


Alpha Band Signal (8-12 Hz) - Subject 06 Baseline
1e−6
Amplitude (µV)
5
0
−5
0
2
4
6
8
10
Time (s)


Alpha Band Signal (8-12 Hz) - Subject 06 Task
1e−6
Amplitude (µV)
5
0
−5
0
2
4
6
8
10
Time (s)


Alpha Band Signal (8-12 Hz) - Subject 07 Baseline
1e−6
Amplitude (µV)
5
0
−5
0
2
4
6
8
10
Time (s)


Alpha Band Signal (8-12 Hz) - Subject 07 Task
1e−6
Amplitude (µV)
5
0
−5
0
2
4
6
8
10
Time (s)

Discussion:

In the alpha band (8-12 Hz) time-domain plots, the baseline conditions for both Subject 06 and Subject 07 show higher amplitudes with more regular and prominent alpha wave oscillations, indicating a relaxed state. During the task condition, the alpha amplitude decreases for both subjects, with less consistent oscillations, reflecting a reduction in alpha power associated with focused mental activity. Notice, Subject 06 shows a more significant reduction in alpha amplitude during the task compared to Subject 07, suggesting a greater shift from relaxation to cognitive engagement. Overall, the task condition results in a noticeable suppression of alpha wave activity, which is more pronounced in Subject 06, indicating stronger task-related involvement.

Code:

```python
import mne
import os
import pandas as pd
from plot_psd import plot_psd
from mne.time_frequency import tfr_multitaper
import numpy as np
import matplotlib.pyplot as plt
from plot_spectogram import plot_spectrogram
from scipy.signal import butter, filtfilt

# Load subject 6 edf files
subject_06_baseline = mne.io.read_raw_edf(os.path.join("data", "Subject06_1.edf"))
subject_06_task = mne.io.read_raw_edf(os.path.join("data", "Subject06_2.edf"))

# Load subject 7 edf files
subject_07_baseline = mne.io.read_raw_edf(os.path.join("data", "Subject07_1.edf"))
subject_07_task = mne.io.read_raw_edf(os.path.join("data", "Subject07_2.edf"))

# Load csv file
csv_file = pd.read_csv(os.path.join("data", "subject-info.csv"))

# Create list of raw data (for easy use)
raw_data = [subject_06_baseline, subject_06_task, subject_07_baseline,
subject_07_task]

# Clean up channels
for raw in raw_data:
   # Rename channels
   raw.rename_channels({
       'EEG Fp1': 'Fp1', 'EEG Fp2': 'Fp2', 'EEG F3': 'F3', 'EEG F4': 'F4',
       'EEG F7': 'F7', 'EEG F8': 'F8', 'EEG T3': 'T3', 'EEG T4': 'T4',
       'EEG C3': 'C3', 'EEG C4': 'C4', 'EEG T5': 'T5', 'EEG T6': 'T6',
       'EEG P3': 'P3', 'EEG P4': 'P4', 'EEG O1': 'O1', 'EEG O2': 'O2',
       'EEG Fz': 'Fz', 'EEG Cz': 'Cz', 'EEG Pz': 'Pz', 'EEG A2-A1': 'A2',
       'ECG ECG': 'ECG'
   })

   # Set channel types for ECG
   raw.set_channel_types({'ECG': 'ecg'})

   # Set the standard montage (10-20 system)
   raw.set_montage(mne.channels.make_standard_montage('standard_1020'))

# Define a function to filter the data in the alpha band (8-12 Hz)
def bandpass_filter(data, sfreq, low_freq, high_freq):
   nyquist = 0.5 * sfreq
   low = low_freq / nyquist
   high = high_freq / nyquist
   b, a = butter(N=4, Wn=[low, high], btype='band')
   return filtfilt(b, a, data)

# List of subjects and their corresponding data
subjects = [
   ('Subject 06 Baseline', subject_06_baseline),
   ('Subject 06 Task', subject_06_task),
   ('Subject 07 Baseline', subject_07_baseline),
   ('Subject 07 Task', subject_07_task)
]

# Create a figure with subplots for each condition
fig, axes = plt.subplots(4, 1, figsize=(15, 20))

# Loop through each subject and condition
for i, (title, raw) in enumerate(subjects):
   # Get data and sampling frequency
   sfreq = raw.info['sfreq']  # Sampling frequency
   data, _ = raw[:]

   # Apply bandpass filter to extract the alpha band (8-12 Hz)
   alpha_data = bandpass_filter(data[0], sfreq, 8, 12)

   # Define a time vector for plotting (using the first 10 seconds)
   time = np.arange(len(alpha_data)) / sfreq

   # Use a subset to make the plot clearer (e.g., first 10 seconds)
   time_subset = time[:int(10 * sfreq)]
   alpha_data_subset = alpha_data[:int(10 * sfreq)]

   # Plot the time domain signal for Alpha band
   axes[i].plot(time_subset, alpha_data_subset, color='blue' if 'Baseline' in title
else 'green')
   axes[i].set_title(f'Alpha Band Signal (8-12 Hz) - {title}', fontsize=14)
   axes[i].set_xlabel('Time (s)', fontsize=10)
```

```
    axes[i].set_ylabel('Amplitude (µV)', fontsize=10)
    axes[i].grid(True)

# Plot data
plt.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.subplots_adjust(hspace=0.7)
plt.show()
```