# Comparative Resting-State EEG Analysis of Alzheimer's Disease

Xueyan Shi, Brandon Chau, Kelvin Li, Jeff Ung

March 20, 2025

## 1  Abstract

This project aims to investigate the differences in electroencephalography (EEG) signals between individuals with Alzheimer's disease (AD) and controls. We used two datapoints from Mitiadous et al. (2023), which contains scalp EEG recordings of AD patients, and healthy controls under eyes-closed rest. In this project, we will first apply signal processing techniques including filtering and artifact rejection remove noise and isolate the frequency range of interest. Our methodology includes the application of the discrete Fourier transform (DFT) to examine spectral components, followed by the computation of power spectral density (PSD) via the Welch method. Through this approach, we aim to quantify the relative band power across different frequency bands, allowing for a detailed comparison of neural activity between the two subjects.

## 2  Exploratory Data Analysis

Below is the dataset description provided by our TA:

> This dataset provides resting-state EEG recordings from individuals with Alzheimer's disease (AD), frontotemporal dementia (FTD), and healthy controls, collected using a clinical EEG system with 19 scalp electrodes during an eyes-closed resting state. The dataset includes 36 AD patients, 23 FTD patients, and 29 healthy age-matched subjects, with Mini-Mental State Examination (MMSE) scores reported for each. EEG signals were recorded using a monopolar montage, and both raw and preprocessed EEG data are available in BIDS format. Preprocessing involved artifact subspace reconstruction and independent component analysis for denoising. This dataset has high reuse potential for studying EEG-based biomarkers for dementia, brain connectivity alterations, and machine learning applications in neurodegenerative disease diagnosis.

- Two individual EEG data included(Alzheimer's disease - subject_001, Healthy Control - subject_027)
- The sampling rate was 500 Hz and the resolution was 10 uV/mm
- Three channel included: `Fp1`, `Fp2`, `F3`

```
[1]: import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     from scipy.signal import butter, firwin, filtfilt, welch, freqz, lfilter

     np.random.seed = 118
```

```
F_s = 500
```

Load in the dataset and transform it into a pandas dataframe.

```
[2]: Control = np.load("./dataset/Control_EEG_sub_027.npy")
     AD = np.load("./dataset/AD_EEG_sub_001.npy")

     feature_names = ['Fp1', 'Fp2', 'F3']
     df_control = pd.DataFrame(Control.T, columns=feature_names)
     df_AD = pd.DataFrame(AD.T, columns=feature_names)

     df_AD.head()
```

```
[2]:         Fp1       Fp2       F3
     0 -0.000190 -0.000142 -0.000107
     1 -0.000180 -0.000137 -0.000100
     2 -0.000167 -0.000135 -0.000106
     3 -0.000160 -0.000133 -0.000105
     4 -0.000159 -0.000124 -0.000104
```

Since the usual measurement for EEG studies is $\mu V$, we upscale the entire dataset by $10^6$

```
[3]: df_AD *= 1e6
     df_control *= 1e6
     df_AD.head()
```

```
[3]:          Fp1         Fp2          F3
     0 -189.892563 -141.845688 -107.373039
     1 -180.419907 -137.353500 -100.048820
     2 -166.992172 -135.058578 -105.761711
     3 -160.205063 -132.958969 -105.322258
     4 -159.326157 -124.462883 -104.150383
```

Display the raw EEG data of two subjects.

```
[4]: def plot_EEG (df,title):
         t = df.index/F_s
         plt.figure(figsize=(12, 4))
         plt.plot(t,df["Fp1"], color = "green", label = "Fp1", alpha = 0.8)
         plt.plot(t,df["Fp2"], color = "orange", label = "Fp2", alpha = 0.8)
         plt.plot(t,df["F3"], color = "blue", label = "F3", alpha = 0.8)
         plt.xlabel('Time (s)')
         plt.ylabel(r'EEG Signal $(\mu V)$')
         plt.title(title)
         plt.legend(title="Channel")
         plt.grid(True, linestyle='--', alpha=0.6)
         plt.show()

     plot_EEG(df_control, "Control EEG")
```

Control EEG
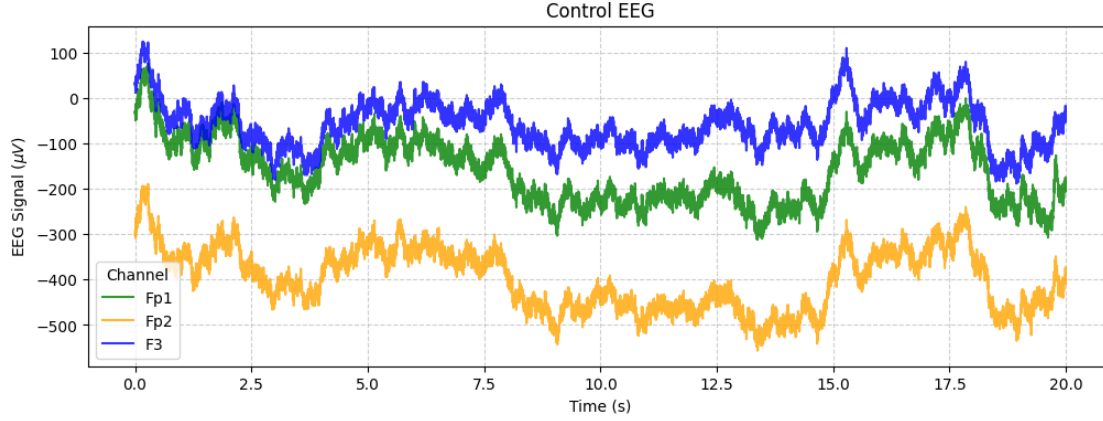
```
[5]: plot_EEG(df_AD, "AD EEG")
```


AD EEG
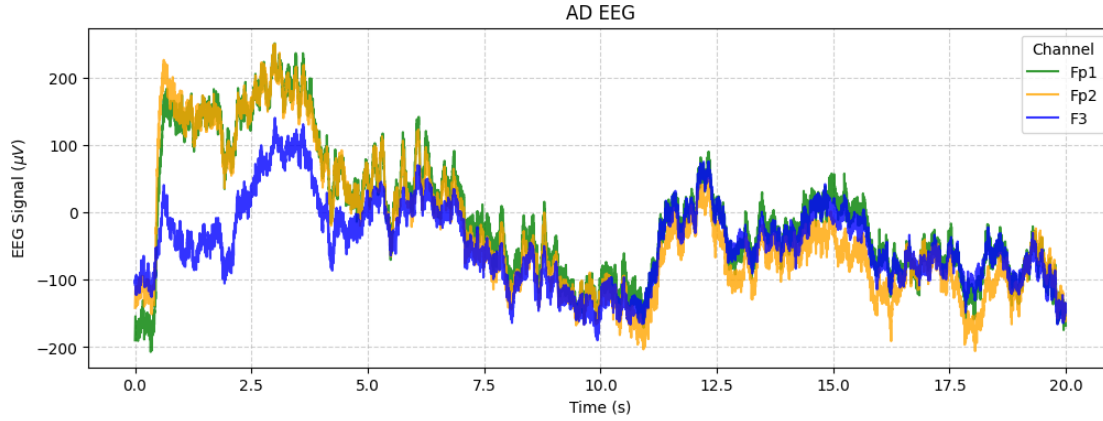
The unprocessed data shows signs of high-frequency noise as the range of the signal varies from $\pm 200 \mu V$, which is abnormal for calm state EEG. Preprocessing and cleaning is needed before doing further analysis.

## 3   Preprocessing

In the original dataset paper, a preprocessing pipeline is applied as the following:

> First, a Butterworth band-pass filter 0.5-45 Hz was applied and the signals were re-referenced to A1-A2. Then, the Artifact Subspace Reconstruction routine (ASR) which is an EEG artifact correction method included in the EEGLab Matlab software was applied to the signals, removing bad data periods which exceeded the max acceptable 0.5 second window standard deviation of 17, which is considered a conservative window. Next, the Independent Component Analysis (ICA) method (RunICA algorithm) was performed, transforming the 19 EEG signals to 19 ICA components. ICA components

3

that were classified as "eye artifacts" or "jaw artifacts" by the automatic classification routine "ICLabel" in the EEGLAB platform were automatically rejected. It should be noted that, even though the recording was performed in a resting state, eyes-closed condition, eye artifacts of eye movement were still found at some EEG recordings.

Since we do not have the whole data set, we will be focusing on the filtering part.

```python
[6]: lowcut = 0.5
     highcut = 45
```

## 3.1   Butterworth Filter

Below is the Butterworth band-pass filter suggested from "scipy-cookbook". It also included a useful graph displaying how the order of butter reflected to the shape of the filter.

```python
[7]: def butter_bandpass(lowcut, highcut, fs, order):
         nyq = 0.5 * fs
         low = lowcut / nyq
         high = highcut / nyq
         b, a = butter(order, [low, high], btype='band')
         return b, a

     def butter_bandpass_filter(data, lowcut, highcut, fs, order):
         b, a = butter_bandpass(lowcut, highcut, fs, order)
         y = filtfilt(b, a, data)
         return y
```

```python
[8]: plt.figure(figsize=(12, 4))

     for order in [3, 6, 9]:
         b, a = butter_bandpass(lowcut, highcut, F_s, order=order)
         w, h = freqz(b, a, worN=2000)
         plt.plot((F_s * 0.5 / np.pi) * w, abs(h), label="order = %d" % order)

     plt.title('IIR Butterworth Bandpass Filters')
     plt.xlabel('Frequency (Hz)')
     plt.ylabel('Gain')
     plt.xlim(0,150)
     plt.grid(True)
     plt.legend(loc='best')
     plt.show()
```
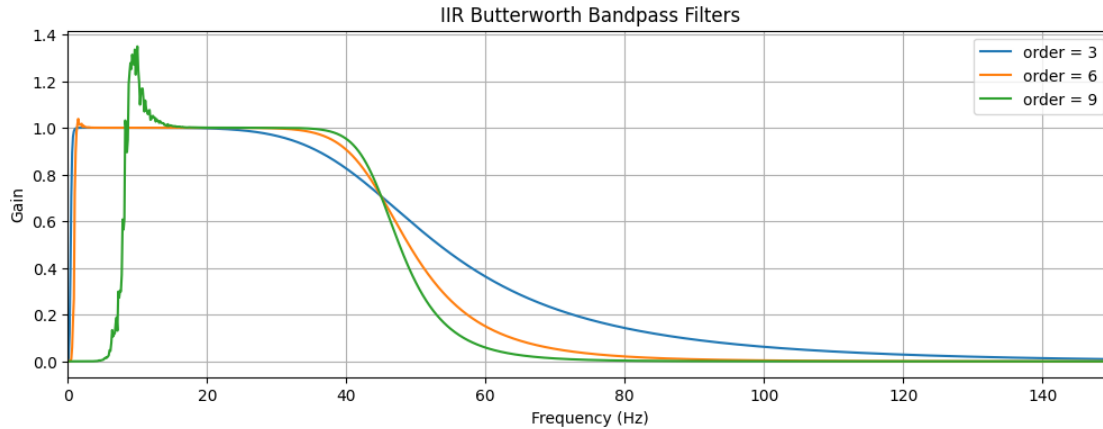
Notice that when `order = 9`, there is a strange spike in the filter. This is likely because of the recursive calculation of the IIR filter. So in our actual filter, we used `order = 5` to prevent this from happening.

```
[9]: df_control_IIRfiltered = df_control.copy()
     df_AD_IIRfiltered = df_AD.copy()

     for col in df_control.columns:
         df_control_IIRfiltered[col] = butter_bandpass_filter(df_control[col],
      ↪lowcut, highcut,F_s, order = 5)
     for col in df_AD.columns:
         df_AD_IIRfiltered[col] = butter_bandpass_filter(df_AD[col], lowcut,
      ↪highcut, F_s, order = 5)


     plot_EEG(df_control_IIRfiltered, "Control Subject's EEG Filtered")
     plot_EEG(df_AD_IIRfiltered, "Alzheimer's Patient's EEG Filtered")
```

Alzheimer's Patient's EEG Filtered
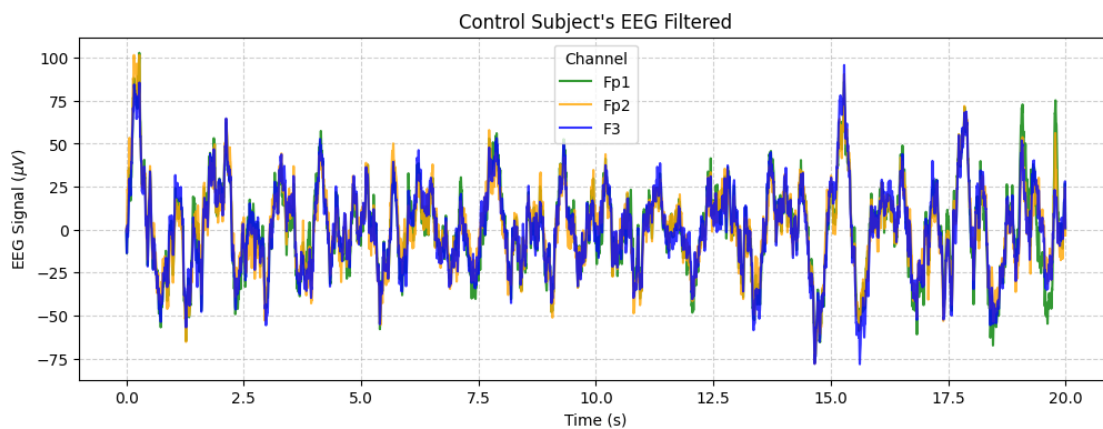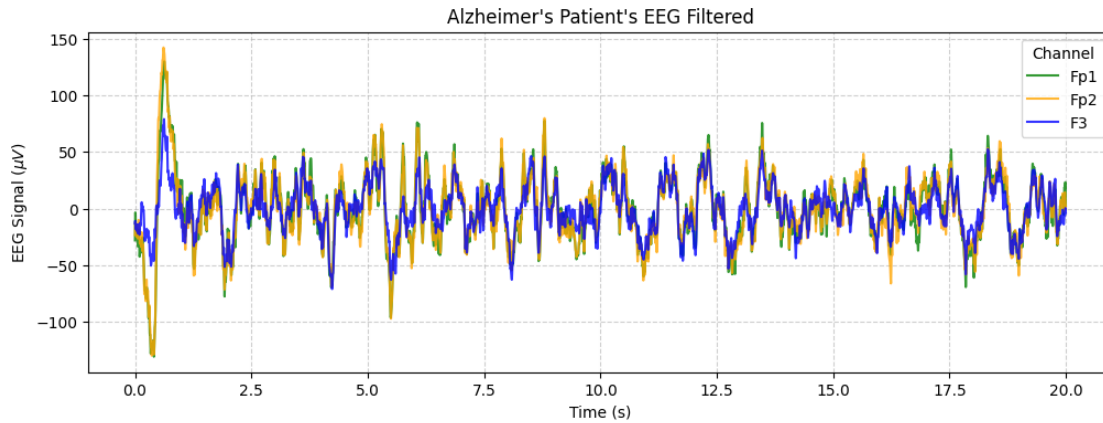
## 3.2 Why Not an IIR Filter

In class we learned a lot a bout finite **impulse response filters** (FIR), but the paper used a **infinite response filter** (IIR). One major reason is because FIR filters need a very high order to achieve the same cutoff as a IIR filter. The original study has 88 subjects with datapoints millions of datapoints, using a high order FIR filter is very inefficient.

But in our study, we only have 2 subjects with 20 seconds of data. This makes it easy for us to use FIR filters without worrying about computation cost. IIR also has a potential disadvantage: since it utilizes feedback loop, without caution one can create significant artifact after applying them. FIR filters only use original data which is safe.

```
[10]: nyquist = F_s / 2
      low = lowcut / nyquist
      high = highcut / nyquist

      plt.figure(figsize=(12, 4))

      for order in [5,10,30,100,300,1000]:
          fir_coeffs = firwin(order, [low,high], pass_zero=False, window="hamming")

          w, h = freqz(fir_coeffs, worN=2000)
          freq = w * F_s / (2 * np.pi)

          plt.plot(freq, abs(h), label=f'order = {order}')

      plt.title('FIR Bandpass Filter with Hamming Window')
      plt.xlabel('Frequency (Hz)')
      plt.xlim(0,150)
      plt.ylabel('Gain')
      plt.grid()
      plt.legend()
      plt.show()
```
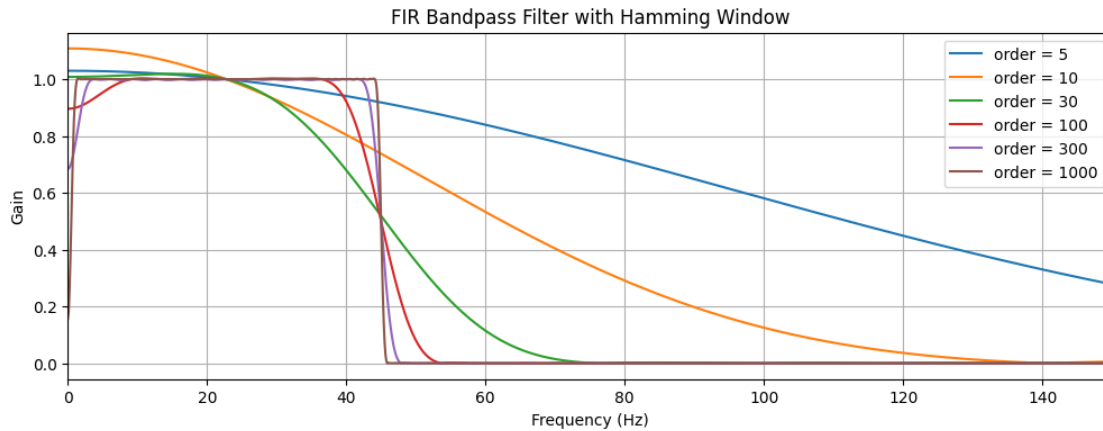
FIR Bandpass Filter with Hamming Window

[11]:
```
order = 1000

fir_coef = firwin(order, [low, high], pass_zero=False, window="hamming")

df_control_FIRfiltered = df_control.copy()
df_AD_FIRfiltered = df_AD.copy()

for col in df_control.columns:
    df_control_FIRfiltered[col] = filtfilt(fir_coef, 1.0, df_control[col])

for col in df_AD.columns:
    df_AD_FIRfiltered[col] = filtfilt(fir_coef, 1.0, df_AD[col])

plot_EEG(df_control_FIRfiltered, "Control's EEG Filtered with 1000 order␣
 ↪Hamming Window")
plot_EEG(df_AD_FIRfiltered, "Alzheimer Patient's EEG Filtered with 1000 order␣
 ↪Hamming Window")
```
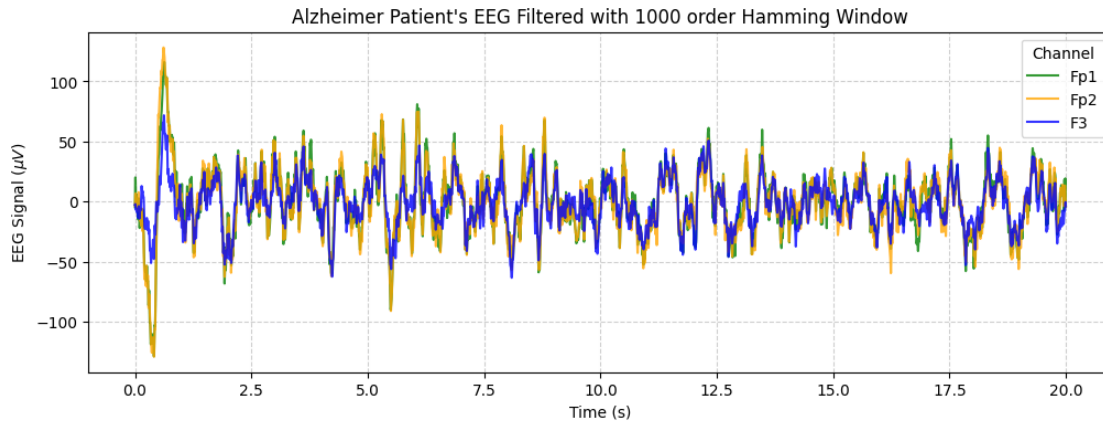


Control's EEG Filtered with 1000 order Hamming Window

Alzheimer Patient's EEG Filtered with 1000 order Hamming Window
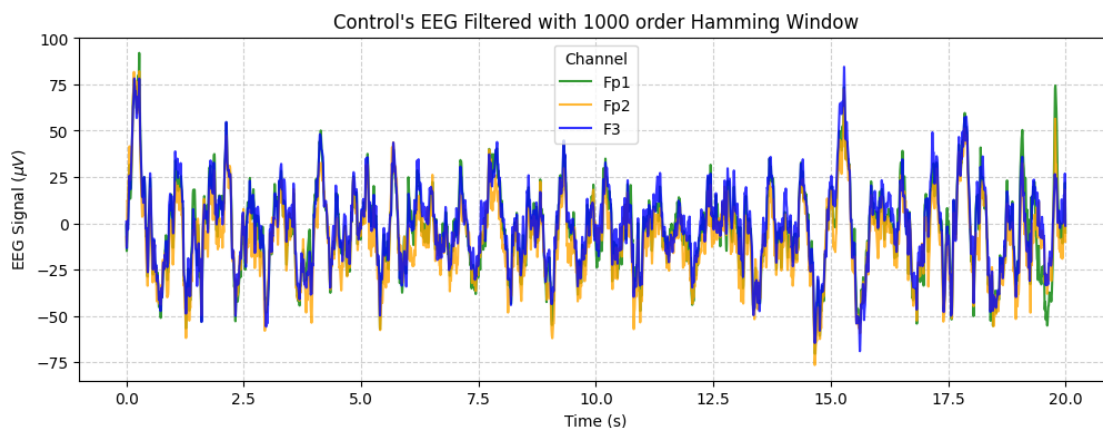
Another problem arise when we have extremely high order: transient effect. Since we only have 20 seconds of data, if we use a FIR response, then for the first few seconds, the filter is still "warming up" and we lost 10% of our data just to initialize the filter itself.

## 4   PSD Analysis

### 4.1   Fourier Transform

We first use the Fast **Fourier Transform (FFT)** which converts time-domain EEG signals into the frequency domain, allowing us to estimate the **Power Spectral Density (PSD)**. FFT-based PSD provides a direct computation with higher frequency resolution which may introduce more variability due to the lack of averaging. Later on, we will be combating this issue by using Welch's method, which averages overlapping segments of the signal to reduce noise, Both methods use the same spectral characteristics, which confirms the validity of our results and ensures that the observed EEG slowing in Alzheimer's patients is a strong finding.

FFT is useful because high-frequency resolution allows us to detect small differences in the EEG power spectrum, which is important for detecting frequency-specific changes in brain activity. For our project, it helps us see that low-frequency power (0-5 Hz) is dominant in Alzheimers patients which reflects EEG slowing. Also, since it provides a detailed spectral decomposition, we can use it to compare power distributions between Alzheimers patients and controls.

First we define the function `compute_psd_fft()`, which calcluates the PSD by applying the FFT to the EEG signal, and then use `plot_PSD_fft()` to visualize the FFT-based PSD for each EEG channel (Fp1, Fp2, F3). This provides a frequency-domain representation of brain activity and allows us to compare spectral differences between Alzheimer's patients and control subjects.

```
[12]: def compute_psd_fft(signal, fs):
          N = len(signal)
          fft_vals = np.fft.rfft(signal)
          fft_freqs = np.fft.rfftfreq(N, d=1/fs)
          psd = (np.abs(fft_vals) ** 2) / N
```

```
        return fft_freqs, psd
```

```
[41]: freqs_Fp1_AD, psd_Fp1_AD = compute_psd_fft(df_AD_FIRfiltered["Fp1"].values, F_s)
      freqs_Fp1_control, psd_Fp1_control =␣
       ↪compute_psd_fft(df_control_FIRfiltered["Fp1"].values, F_s)

      freqs_Fp2_AD, psd_Fp2_AD = compute_psd_fft(df_AD_FIRfiltered["Fp2"].values, F_s)
      freqs_Fp2_control, psd_Fp2_control =␣
       ↪compute_psd_fft(df_control_FIRfiltered["Fp2"].values, F_s)

      freqs_F3_AD, psd_F3_AD = compute_psd_fft(df_AD_FIRfiltered["F3"].values, F_s)
      freqs_F3_control, psd_F3_control = compute_psd_fft(df_control_FIRfiltered["F3"].
       ↪values, F_s)

      fig, axes = plt.subplots(3, 1, figsize=(8, 8), sharex=True)

      # Fp1
      axes[0].plot(freqs_Fp1_AD, psd_Fp1_AD, label="Alzheimer", color="red", alpha=0.
       ↪8)
      axes[0].plot(freqs_Fp1_control, psd_Fp1_control, label="Control", color="blue",␣
       ↪alpha=0.8)
      axes[0].set_ylabel(r"PSD ($muV^2$/Hz)")
      axes[0].set_title("Power Spectral Density - Fp1")
      axes[0].legend()
      axes[0].grid(linestyle="--", alpha=0.6)
      axes[0].set_yscale('log')
      axes[0].axvspan(0.5, 4, color='green', alpha=0.3, label='Delta Band')
      axes[0].axvspan(8, 13, color='yellow', alpha=0.3, label='Alpha Band')

      # Fp2
      axes[1].plot(freqs_Fp2_AD, psd_Fp2_AD, label="Alzheimer", color="red", alpha=0.
       ↪8)
      axes[1].plot(freqs_Fp2_control, psd_Fp2_control, label="Control", color="blue",␣
       ↪alpha=0.8)
      axes[1].set_ylabel(r"PSD ($muV^2$/Hz)")
      axes[1].set_title("Power Spectral Density - Fp2")
      axes[1].legend()
      axes[1].grid(linestyle="--", alpha=0.6)
      axes[1].set_yscale('log')
      axes[1].axvspan(0.5, 4, color='green', alpha=0.3, label='Delta Band')
      axes[1].axvspan(8, 13, color='yellow', alpha=0.3, label='Alpha Band')

      # F3
      axes[2].plot(freqs_F3_AD, psd_F3_AD, label="Alzheimer", color="red", alpha=0.8)
      axes[2].plot(freqs_F3_control, psd_F3_control, label="Control", color="blue",␣
       ↪alpha=0.8)
```
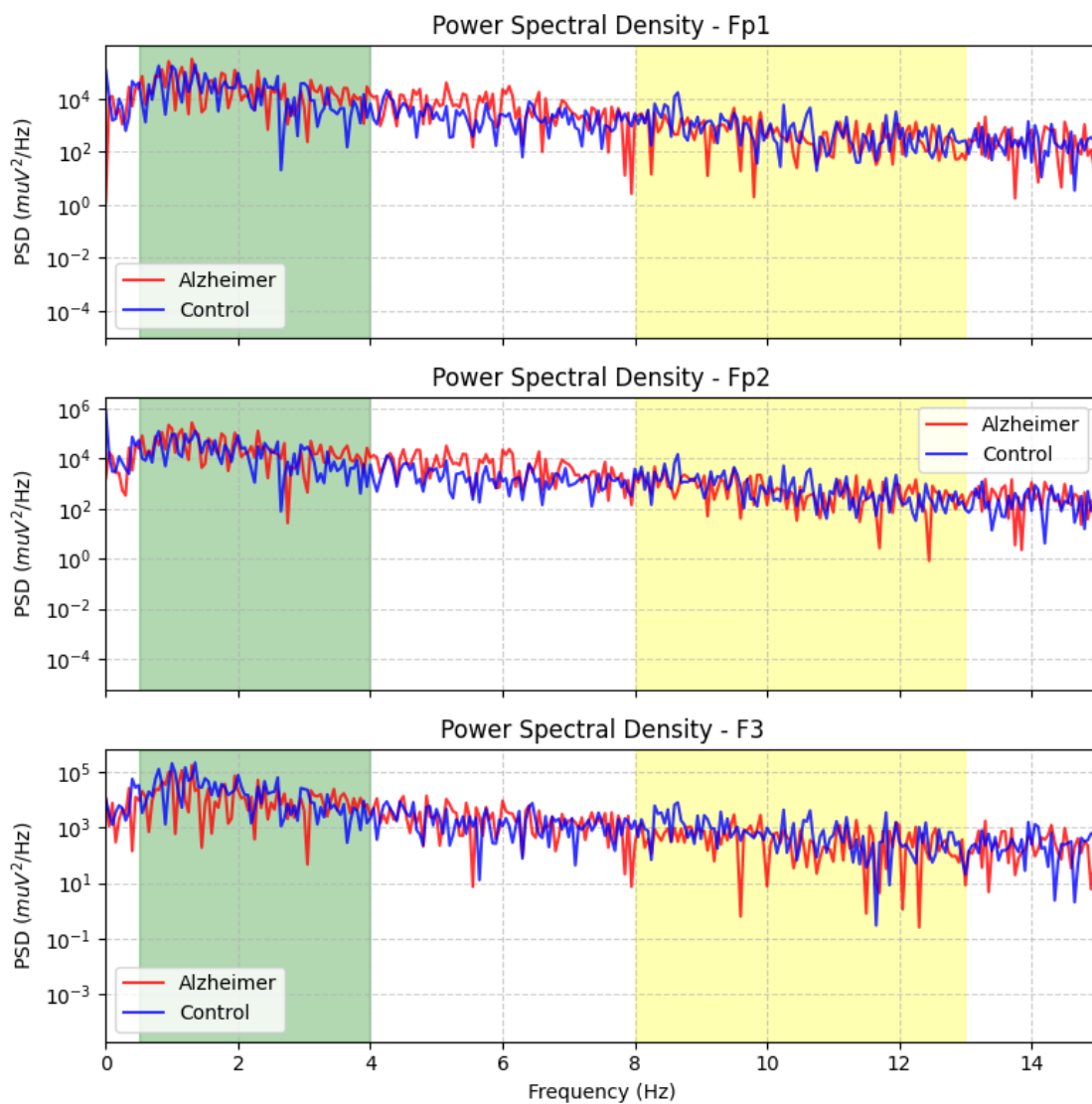
```python
axes[2].set_xlabel("Frequency (Hz)")
axes[2].set_ylabel(r"PSD ($muV^2$/Hz)")
axes[2].set_title("Power Spectral Density - F3")
axes[2].legend()
axes[2].grid(linestyle="--", alpha=0.6)
axes[2].set_yscale('log')
axes[2].axvspan(0.5, 4, color='green', alpha=0.3, label='Delta Band')
axes[2].axvspan(8, 13, color='yellow', alpha=0.3, label='Alpha Band')

plt.xlim(0, 15)
plt.tight_layout()
plt.show()
```

## 4.2  Welch's Method

To analyze the and compare the EEG signals from Alzheimer's patients and control subjects, we do a Power Spectral Density (PSD) analysis. This will show us how power is distributed across different frequencies, revealing key features related to brain activity. To estimate the PSD, we use **Welch's Method**, which reduces noise and provides a smoother estimate by averaging multiple segments of the signal. This allows us to compare the power across different frequency bands while minimizing variability.

We'll first create a function to calculate the PSD and plot the PSD for all 3 EEG channels (`Fp1`, `Fp2`, `F3`).

```python
[37]: def compute_and_plot_psd(df_AD, df_control, nperseg=500):
          freqs_Fp1_AD, psd_Fp1_AD = welch(df_AD["Fp1"].values, fs=F_s,
      ↪nperseg=nperseg)
          freqs_Fp1_control, psd_Fp1_control = welch(df_control["Fp1"].values,
      ↪fs=F_s, nperseg=nperseg)

          freqs_Fp2_AD, psd_Fp2_AD = welch(df_AD["Fp2"].values, fs=F_s,
      ↪nperseg=nperseg)
          freqs_Fp2_control, psd_Fp2_control = welch(df_control["Fp2"].values,
      ↪fs=F_s, nperseg=nperseg)

          freqs_F3_AD, psd_F3_AD = welch(df_AD["F3"].values, fs=F_s, nperseg=nperseg)
          freqs_F3_control, psd_F3_control = welch(df_control["F3"].values, fs=F_s,
      ↪nperseg=nperseg)

          fig, axes = plt.subplots(3, 1, figsize=(8, 8), sharex=True)

          for i, (freqs_AD, psd_AD, freqs_control, psd_control, title) in enumerate([
              (freqs_Fp1_AD, psd_Fp1_AD, freqs_Fp1_control, psd_Fp1_control, "Fp1"),
              (freqs_Fp2_AD, psd_Fp2_AD, freqs_Fp2_control, psd_Fp2_control, "Fp2"),
              (freqs_F3_AD, psd_F3_AD, freqs_F3_control, psd_F3_control, "F3")
          ]):
              axes[i].plot(freqs_AD, psd_AD, label="Alzheimer", color="red", alpha=0.
      ↪8)
              axes[i].plot(freqs_control, psd_control, label="Control", color="blue",
      ↪alpha=0.8)
              axes[i].set_ylabel(r"PSD ($\mu V^2$/Hz)")
              axes[i].set_title(f"Power Spectral Density - {title}")
              axes[i].legend()
              axes[i].grid(linestyle="--", alpha=0.6)
              axes[i].set_yscale('log')
              axes[i].axvspan(0.5, 4, color='green', alpha=0.3, label='Delta Band')
              axes[i].axvspan(8, 13, color='yellow', alpha=0.3, label='Alpha Band')

          plt.xlim(0, 15)
          plt.xlabel("Frequency (Hz)")
```

```
    plt.tight_layout()
    plt.show()
```

We want to determine what `nperseg` value will best help us determine our problem. Testing different `nperseg` values is essential because it affects two important properties of the PSD: **Frequency Resolution** and **Variance Reduction**. A larger `nperseg` will increase frequency resolution (i.e distinguish fine details in the EEG spectrum) as well as reduce number of segments, making the estimate more variable. While a smaller `nperseg` will be the vice versa.

First, we make `nperseg` the same as the sampling rate (**500**).

```
[38]: compute_and_plot_psd(df_AD_FIRfiltered, df_control_FIRfiltered, nperseg=500)
```
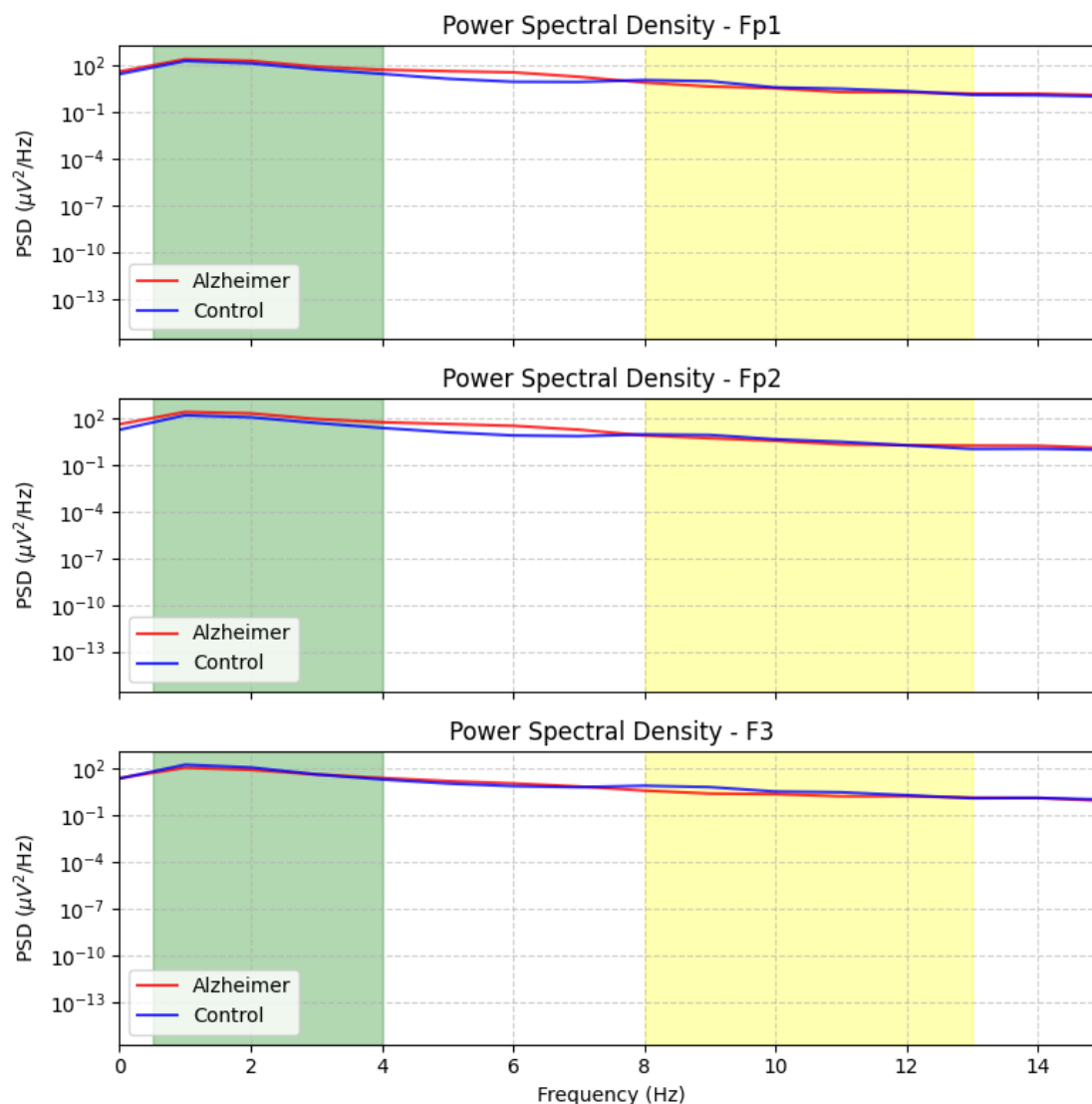


You can see that the PSD curve is smooth with reduced noise but the **frequency resolution** is

low, so it is harder to distinguish small peaks and be definitive with our conclusion. The peaks in **Delta (Green Band)** and **Alpha (Yellow Band)** bands are less distinct.

Next, we try using a `nperseg` value of 1500.

[39]: `compute_and_plot_psd(df_AD_FIRfiltered, df_control_FIRfiltered, nperseg=1500)`
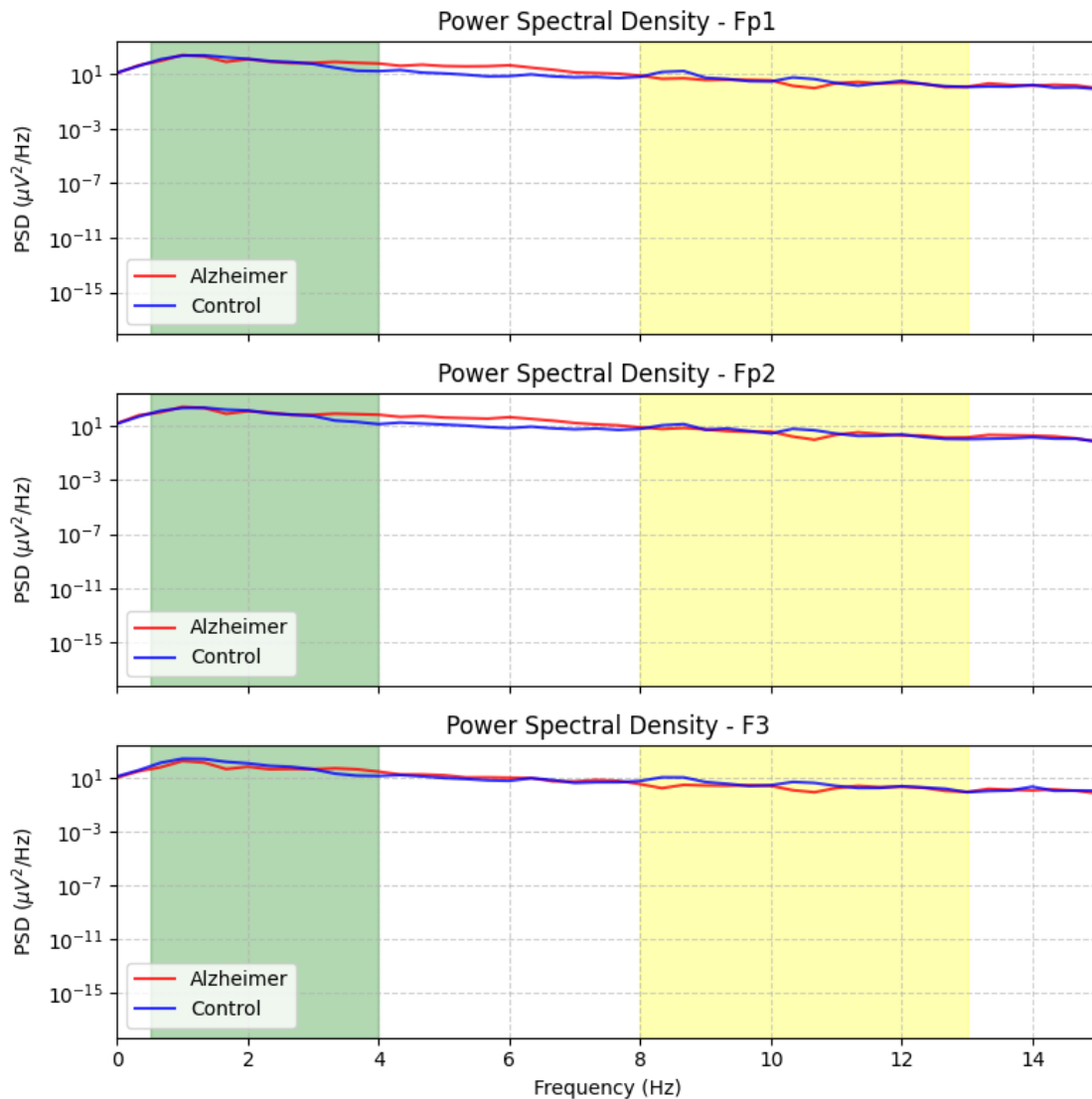


As you can observe, there looks to be a balance between **frequency resolution** and **smoothness**. Even though the PSD still has some averaging effect, the individual peaks are more visible. This allows us to clearly see the peaks in the **Delta** and **Alpha** bands.

Lastly, we try using a high `nperseg` value of 2500.

[40]: `compute_and_plot_psd(df_AD_FIRfiltered, df_control_FIRfiltered, nperseg=2500)`

Power Spectral Density - Fp1

Power Spectral Density - Fp2

Power Spectral Density - F3
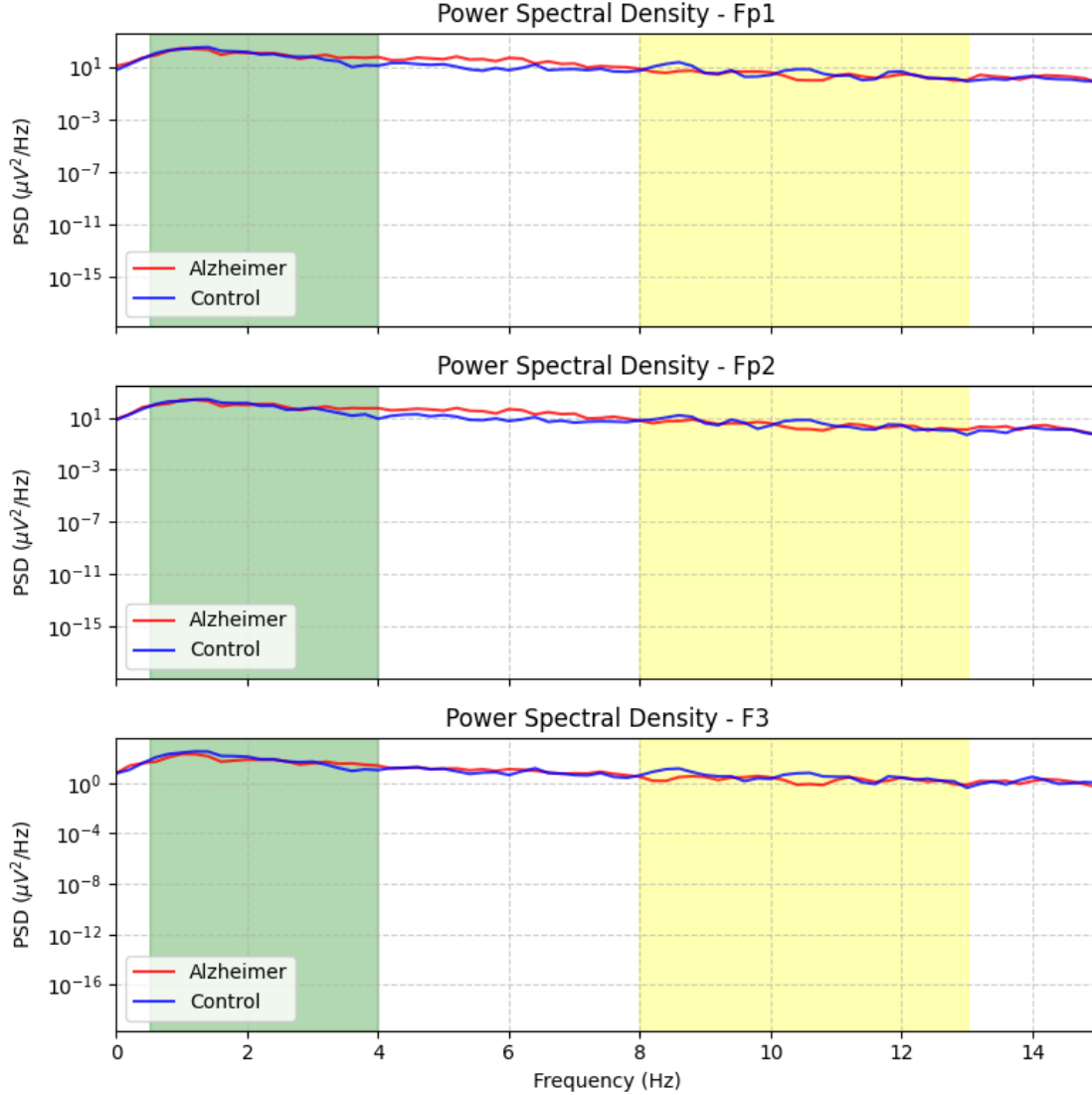
You can see that there is high **frequency resolution** but the curve looks **more noisier** since there are fewer segments to average over. We can observe that the peaks in the **Delta** and **Alpha** bands are more sharp.

However, we chose to stick with the value of 1500 because there was **sufficient frequency resolution** to detect key EEG oscillations along with **minimal noise** without over smoothing the PSD.

## 5  Conclusion

Our preliminary analysis of representative EEG recordings from the Miltiadous et al. (2023) dataset reveals that the healthy control's EEG consistently exhibits a prominent alpha peak in the 8–12 Hz range, whereas the EEG from an Alzheimer's disease (AD) patient shows a significantly attenuated

alpha-band component with a relative shift toward lower frequencies. When compared with the group-level analyses presented in the Miltiadous et al. paper—derived from a large dataset of 88 subjects—there is a clear convergence of findings. The paper demonstrates that AD patients have reduced relative alpha power compared to healthy controls; after rigorous preprocessing and spectral estimation (using techniques such as Welch's method to compute the power spectral density), AD patients consistently show a decrease in alpha-band power and, in some cases, an increase in slower rhythms (e.g., theta power), reflecting diminished neural synchrony.

In our analysis, the healthy control's spectrum shows a well-defined alpha peak that aligns with the typical EEG signature of healthy resting-state activity, while the AD subject's spectrum lacks a robust alpha peak, suggesting that the neural networks responsible for generating these oscillations may be compromised. These observations mirror the global pattern seen in the paper, where the relative alpha power is consistently lower in the AD group across the entire dataset—a phenomenon widely documented in the literature as a hallmark of Alzheimer's disease, reinforcing this association.

Furthermore, the similarity between our representative analysis and the paper's group-level results indicates that the frequency band differences are not isolated occurrences but reflect a robust, global alteration in the EEG dynamics of AD patients. This supports the potential of alpha-band power as a reliable non-invasive biomarker for Alzheimer's disease. Extending our analysis to the full dataset would likely reveal statistically significant differences in alpha power between AD patients and healthy controls, as demonstrated by the comprehensive heatmaps and classification benchmarks in the paper. Overall, our initial findings, together with the published results, support the hypothesis that reduced alpha-band power is a global feature of the resting-state EEG in Alzheimer's disease.