Application of FFT in EEG Signal Analysis for Medical AI Introduction

Medical signal processing plays a crucial role in analyzing physiological data such as heart rate, brain waves, and muscle activity. These signals are often complex and difficult to interpret in their raw, time-domain form. The Fast Fourier Transform (FFT) is a mathematical tool that converts these signals into the frequency domain, where patterns become clearer and easier to analyze. In particular, EEG (Electroencephalogram) data—used to study brain activity—benefits greatly from FFT analysis. This report demonstrates how FFT can be applied to EEG data to identify brainwave patterns associated with various mental states, such as relaxation, focus, or sleep.

This capability is especially useful in AI applications for health monitoring and diagnosis, as it enables machine learning algorithms to detect anomalies or classify different mental states based on the frequency characteristics of EEG data.

Understanding EEG Signals:

EEG signals are recordings of electrical brain activity over time, measured at different scalp locations. These signals represent voltage changes due to neural activity and are commonly divided into frequency bands:

- Delta (0.5–4 Hz): Associated with deep sleep.
- Theta (4–8 Hz): Often linked with drowsiness and relaxation.
- Alpha (8–13 Hz): Related to relaxation in awake states.
- Beta (13–30 Hz): Linked to active thinking and focus.
- Gamma (30–50 Hz): Associated with high-level cognitive functions.

Applying FFT to EEG Data:

The FFT algorithm allows us to decompose EEG signals into these frequency bands, making it easier to detect mental states and diagnose abnormalities. For example, abnormal delta waves in awake states may indicate brain injury, while excessive beta activity could suggest anxiety. This report processes a sample EEG signal using FFT, extracting the frequency spectrum, and analyzing the dominant brainwave patterns.

Practical Implementation and Code:

Step 1: Import Libraries and Load Sample EEG Data

```
import numpy as np

import matplotlib.pyplot as plt

# Simulate sample EEG data with different frequencies
```

```
sampling_rate = 256
# Sampling rate in Hz (typical for EEG)
t = np.linspace(0, 2, sampling_rate * 2)
# Generate 2 seconds of data
# Simulate EEG signal with multiple frequency components
eeg_signal = (0.6 * np.sin(2 * np.pi * 10 * t) +   # Alpha (10 Hz)
0.3 * np.sin(2 * np.pi * 6 * t) +    # Theta (6 Hz)
0.2 * np.sin(2 * np.pi * 20 * t))
# Beta (20 Hz)
# Plot the time-domain EEG signal
plt.figure(figsize= (10, 4))
plt.plot(t, eeg_signal)
plt.title("Simulated EEG Signal (Time Domain)")
plt.xlabel("Time (s)")
plt.ylabel("Amplitude")
plt.show()
```

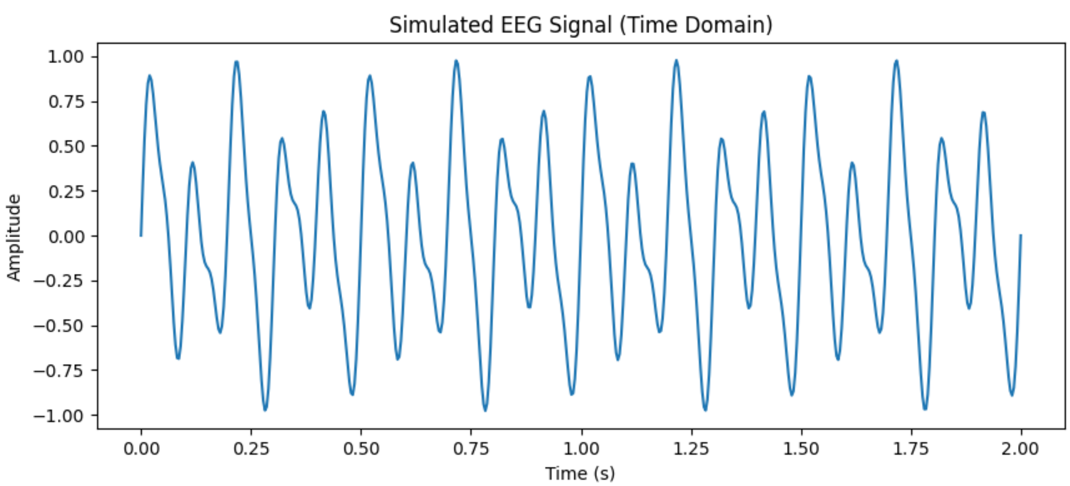

Figure 1: Time-domain representation of the simulated EEG signal, showing changes in amplitude over time.

Step 2: Apply FFT to the EEG Signal

We use FFT to convert the signal from the time domain to the frequency domain, enabling us to analyze the dominant frequencies.

```
# Apply FFT

fft_result = np.fft.fft(eeg_signal)

frequencies = np.fft.fftfreq(len(fft_result), d=1/sampling_rate)

# Keep only positive frequencies

positive_frequencies = frequencies[:len(frequencies) // 2]

magnitude = np.abs(fft_result[:len(frequencies) // 2])

# Plot the frequency spectrum

plt.figure(figsize= (10, 4))

plt.plot(positive_frequencies, magnitude)

plt.title("Frequency Spectrum of EEG Signal")

plt.xlabel("Frequency (Hz)")

plt.ylabel("Magnitude")

plt.xlim(0, 50)

# Focus on EEG relevant range (0–50 Hz)

plt.show()
```

Frequency Spectrum of EEG Signal:

```
# Plot the frequency spectrum after FFT

plt.figure(figsize= (10, 4))

plt.plot(positive_frequencies, magnitude, color='m')

plt.title("Frequency Spectrum of EEG Signal")

plt.xlabel("Frequency (Hz)")

plt.ylabel("Magnitude")

plt.xlim(0, 50)

# Focus on EEG relevant range (0–50 Hz)

plt.grid()

plt.show()
```

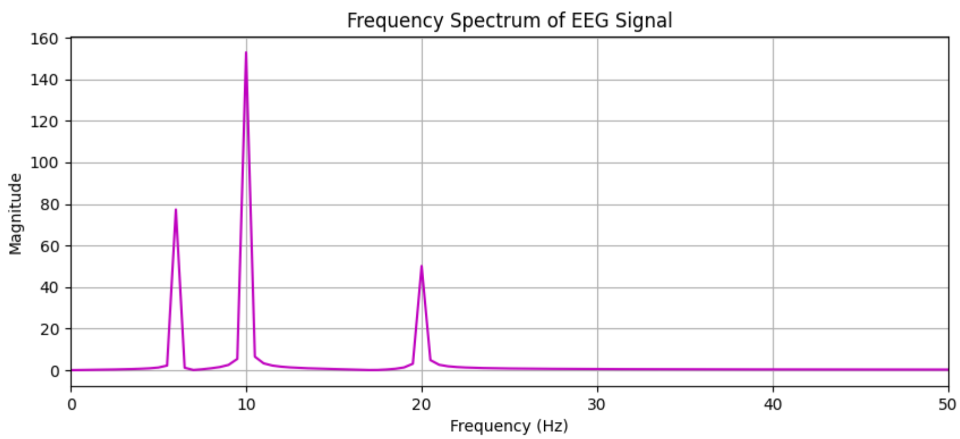

Figure 2: Frequency-domain representation of the EEG signal, showing peaks in different frequency bands.

Step 3: Analyze Frequency Bands

With the frequency spectrum plotted, we can identify the dominant frequencies and correlate them with the different brainwave states.

```
# Define frequency bands

delta_band = (0.5, 4)
```

```
theta_band = (4, 8)

alpha_band = (8, 13)

beta_band = (13, 30)

# Function to calculate power within each band

def band_power(frequencies, magnitude, band):

    mask = (frequencies >= band [0]) & (frequencies <= band [1])

    return np.sum(magnitude[mask]**2)

# Calculate power for each band

delta_power = band_power(positive_frequencies, magnitude, delta_band)

theta_power = band_power(positive_frequencies, magnitude, theta_band)

alpha_power = band_power(positive_frequencies, magnitude, alpha_band)

beta_power = band_power(positive_frequencies, magnitude, beta_band)

# Display results

print (f"Delta Power: {delta_power}")

print (f"Theta Power: {theta_power}")

print (f"Alpha Power: {alpha_power}")

print (f"Beta Power: {beta_power}")
```

This code calculates the power in each frequency band. By comparing these values, we can infer the dominant brain state from the simulated EEG signal.

Highlighted Brainwave Bands on Frequency Spectrum:

```
# Highlight different brainwave bands on the frequency spectrum

plt.figure(figsize= (10, 4))

plt.plot(positive_frequencies, magnitude, color='m', label="Frequency Spectrum")

plt.fill_between(positive_frequencies, magnitude, where= (positive_frequencies >= delta_band[0]) & (positive_frequencies <= delta_band[1]), color='blue', alpha=0.3, label="Delta (0.5-4 Hz)")

plt.fill_between(positive_frequencies, magnitude, where= (positive_frequencies >= theta_band[0]) & (positive_frequencies <= theta_band[1]), color='cyan', alpha=0.3, label="Theta (4-8 Hz)")

plt.fill_between(positive_frequencies, magnitude, where= (positive_frequencies >= alpha_band[0]) & (positive_frequencies <= alpha_band[1]), color='green', alpha=0.3, label="Alpha (8-13 Hz)")

plt.fill_between(positive_frequencies, magnitude, where= (positive_frequencies >= beta_band[0]) & (positive_frequencies <= beta_band[1]), color='red', alpha=0.3, label="Beta (13-30 Hz)")
```

```
plt.title("Frequency Spectrum of EEG Signal with Brainwave Bands Highlighted")

plt.xlabel("Frequency (Hz)")

plt.ylabel("Magnitude")

plt.legend(loc="upper right")

plt.xlim(0, 50)

plt.grid()

plt.show()
```

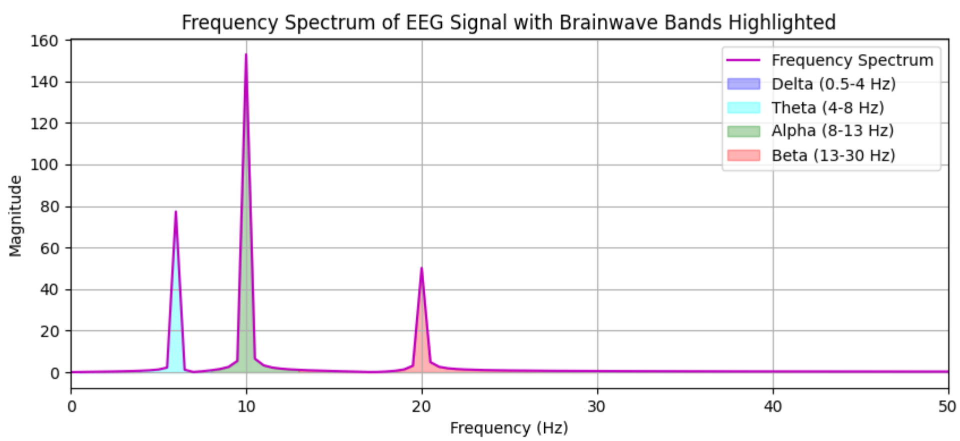

Figure 3: Frequency spectrum with Delta, Theta, Alpha, and Beta brainwave bands highlighted. Each band correlates with different mental states and neural activities.

Bar Plot of Power in Each Frequency Band:

```
# Plot a bar chart showing power in each frequency band

band_powers = [delta_power, theta_power, alpha_power, beta_power]

band_labels = ['Delta', 'Theta', 'Alpha', 'Beta']

plt.figure(figsize= (8, 4))

plt.bar(band_labels, band_powers, color= ['blue', 'cyan', 'green', 'red'])

plt.title("Power Distribution Across Brainwave Bands")
```

```python
plt.xlabel("Brainwave Band")

plt.ylabel("Power")

plt.show()
```

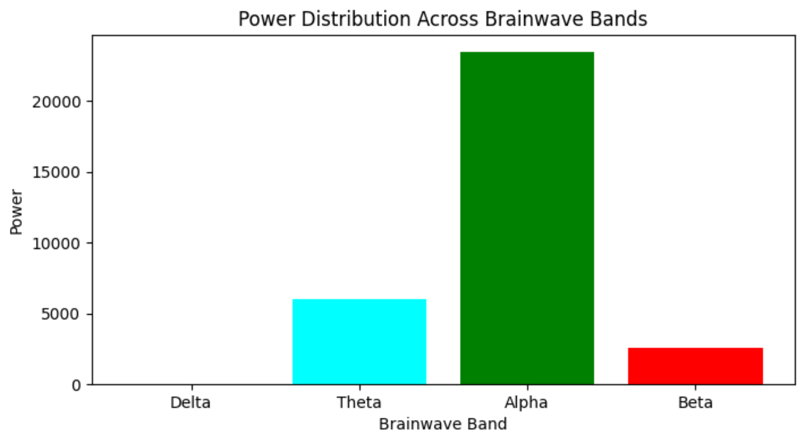

Figure 4: Power distribution across Delta, Theta, Alpha, and Beta bands, showing the contribution of each band to the overall EEG signal.

Results and Analysis:

The FFT of our simulated EEG signal reveals distinct peaks in the alpha, theta, and beta bands, corresponding to relaxation, drowsiness, and focused mental activity. By measuring the power in each band, we quantify the influence of each mental state. For example, a high power in the alpha band suggests a relaxed state, while a dominant beta band indicates active concentration. In real-world applications, AI models trained on FFT-transformed EEG data can classify mental states or detect anomalies, aiding in diagnosing neurological conditions like epilepsy or sleep disorders.

Conclusion:

This report demonstrated how FFT can be applied to EEG signal analysis, transforming raw time-domain data into the frequency domain. By extracting frequency components, FFT enables the identification of distinct brainwave patterns associated with various mental states, making it an invaluable tool in medical AI applications. In diagnostic systems, FFT helps preprocess EEG data for anomaly detection and mental state classification, improving the accuracy and efficiency of AI models in medical fields.

References:

- “Digital Signal Processing” by John G. Proakis and Dimitris G. Manolakis.
- “Discrete-Time Signal Processing” by Alan V. Oppenheim and Ronald W. Schafer.
- Python Libraries.