Name: Tanisha Pal **Registration No: Intern**

Neuroinformatic **Date: 18/01/24**

DIGITAL ASSIGNMENT 1

Question 1:

The linked 2*300 Matlab array stores the spectral power obtained from a test subject participating in a memory game for a certain duration of interest. The columns represent the successful and unsuccessful conditions, respectively; the rows represent the trials.

Evaluate if the observed power is significantly different across the two conditions and report the results. Justify your choice of statistical method in a line or two

Answer 1:

Here we are using a paired-sample t-test to assess whether the observed power is significantly different across two conditions (successful and unsuccessful). The paired-sample t-test, also known as the dependent-sample t-test, is a statistical test used to determine whether there is a significant difference between the means of two related groups. It is typically applied when the data involves pairs of observations or repeated measurements on the same subjects or items.

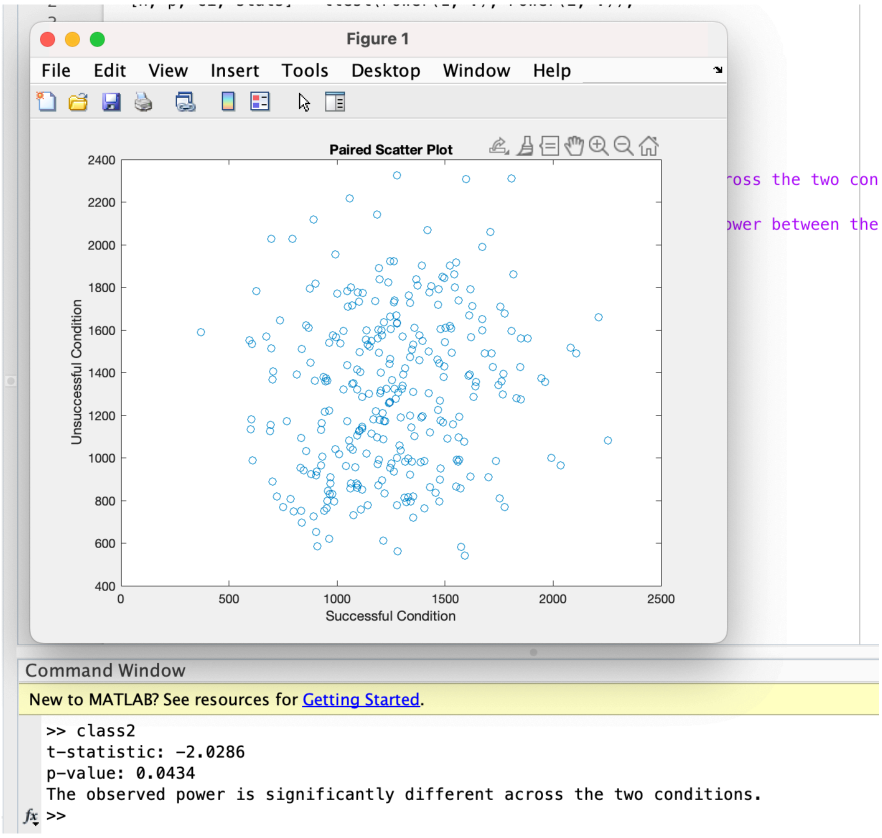
In our case, we have paired data because each subject has both successful and unsuccessful conditions measured. The paired-sample t-test is appropriate for analysing paired data, where each observation in one condition is paired with a related observation in the other condition.

```
load('/Users/tanisha/Desktop/IIITH COGSC/NEUROINFORMATICS/NEURO LAB/power.mat');
[h, p, ci, stats] = ttest(Power(1, :), Power(2, :));

% Display the results
fprintf('t-statistic: %.4f\n', stats.tstat);
fprintf('p-value: %.4f\n', p);

% Check if the difference is statistically significant
if h
    fprintf('The observed power is significantly different across the two conditions.\n');
else
    fprintf('There is no significant difference in observed power between the two conditions.\n');
end

% Paired Scatter Plot
figure;
plot(Power(1, :), Power(2, :), 'o');
xlabel('Successful Condition');
ylabel('Unsuccessful Condition');
title('Paired Scatter Plot');
```



Question 2:

Create four individual sine waves and a mixed wave.
Take four frequencies: 2 Hz, 8 Hz, 12 Hz, and 25 Hz. Create four sine waves of 4 s duration, sampled every 1 ms. Choose different amplitudes and different phase lags of your choice.
[Hint: It's the Class XII physics formula; nothing fancy but feel free to Google.]
Now just average the four waves to get the mixed wave.
Plot the four individual waves and the mixed wave in a 5*1 panel.
Can you recognize the involved frequencies in the mixed wave from the plot?
If not, can you do the necessary processing to recover this information (use Google)?

Answer 2:

```
% Parameters for individual sine waves
duration = 4;  % seconds
sampling_rate = 1000;  % 1 ms sampling interval
time = 0:1/sampling_rate:duration;

% Frequencies of the sine waves
frequencies = [2, 8, 12, 25];

% Amplitudes and phase lags for each sine wave
amplitudes = [1, 0.8, 0.5, 0.3];
phase_lags = [pi/4, pi/2, pi, 3*pi/4];

% Initialize the mixed wave
mixed_wave = zeros(size(time));
```

```
% Generate individual sine waves and compute mixed wave
figure;

for i = 1:4
    % Generate individual sine wave
    individual_wave = amplitudes(i) * sin(2 * pi * frequencies(i) * time +
phase_lags(i));

    % Plot individual wave
    subplot(5, 1, i);
    plot(time, individual_wave);
    title(['Individual Wave ', num2str(i)]);

    % Accumulate for mixed wave
    mixed_wave = mixed_wave + individual_wave;
end

% Average the individual waves to get the mixed wave
mixed_wave = mixed_wave / 4;

% Plot mixed wave
subplot(5, 1, 5);
plot(time, mixed_wave);
title('Mixed Wave');

xlabel('Time (s)');
ylabel('Amplitude');
sgtitle('Individual and Mixed Sine Waves');

% Create a new figure (figure 2)
figure;

% Plot the mixed wave
subplot(2, 1, 1);
plot(time, mixed_wave);
title('Mixed Wave');
xlabel('Time (s)');
ylabel('Amplitude');

% Perform Fourier transform to decompose into individual frequencies
fft_result_mixed = fft(mixed_wave);
frequency_mixed = sampling_rate * (0:(length(mixed_wave) / 2)) /
length(mixed_wave);

% Plot the frequency spectrum of the mixed wave
subplot(2, 1, 2);
plot(frequency_mixed, abs(fft_result_mixed(1:length(frequency_mixed))));
title('Frequency Spectrum of Mixed Wave');
xlabel('Frequency (Hz)');
ylabel('Amplitude');
xlim([0 30]);
```

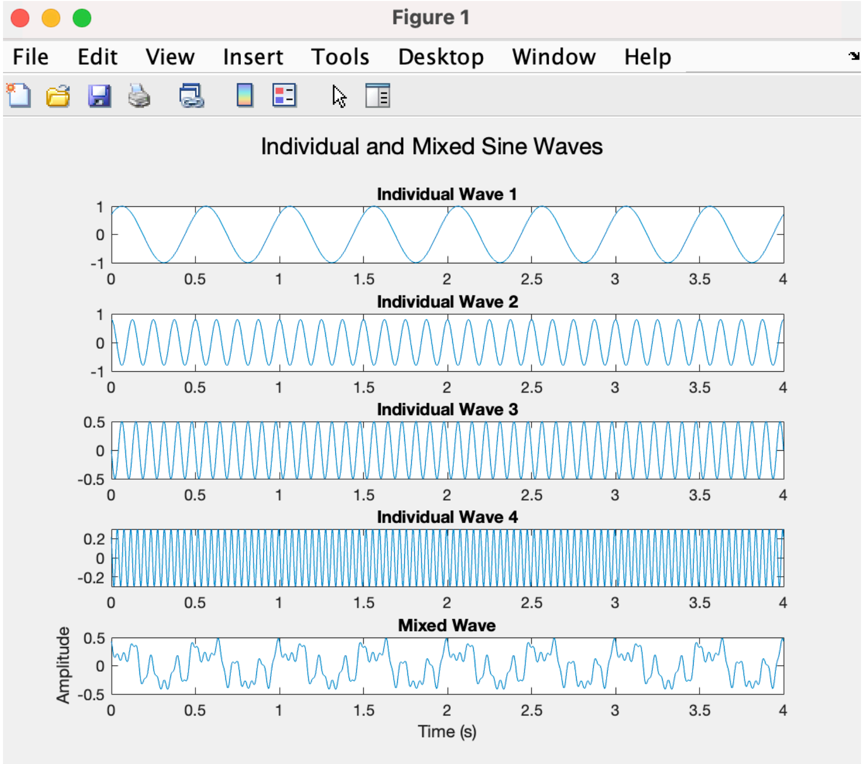


Figure 1
File Edit View Insert Tools Desktop Window Help
Individual and Mixed Sine Waves
Individual Wave 1
Individual Wave 2
Individual Wave 3
Individual Wave 4
Mixed Wave
Amplitude
Time (s)

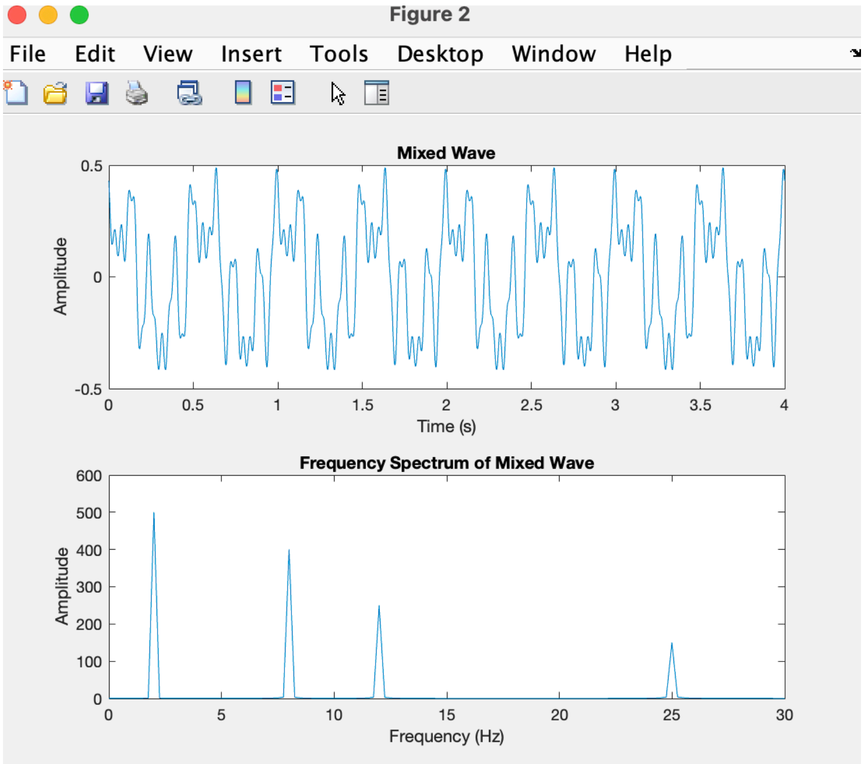


Figure 2
File Edit View Insert Tools Desktop Window Help
Mixed Wave
Amplitude
Time (s)
Frequency Spectrum of Mixed Wave
Amplitude
Frequency (Hz)

Question 3:

To all of the individual waves, add different amounts of white noise, plot, and redo the frequency recovery

Answer 3:
We are taking 4 sine waves of frequencies(2, 8, 12, 25) and adding white noise levels(0.1, 0.5, 1, 2)

```
% MATLAB code for generating sine waves, adding white noise, and frequency
recovery

% Define parameters
duration = 4;  % Duration of the signal in seconds
sampling_rate = 1000;  % Sampling rate in Hz
time = 0:1/sampling_rate:duration-1/sampling_rate;  % Time vector

% Frequencies of sine waves
frequencies = [2, 8, 12, 25];  % in Hz

% Generate sine waves
sine_waves = zeros(length(frequencies), length(time));

for i = 1:length(frequencies)
    sine_waves(i, :) = sin(2 * pi * frequencies(i) * time);
end

% Add white noise
noise_levels = [0.1, 0.5, 1, 2];  % Noise levels to be added

for i = 1:length(frequencies)
    noise = noise_levels(i) * randn(1, length(time));  % White noise
    sine_waves(i, :) = sine_waves(i, :) + noise;  % Add white noise
end

% Plot the signals
figure;

for i = 1:length(frequencies)
    subplot(length(frequencies), 1, i);
    plot(time, sine_waves(i, :));
    title(['Signal with White Noise (Frequency: ' num2str(frequencies(i))
' Hz)']);
    xlabel('Time (s)');
    ylabel('Amplitude');
end

% Frequency recovery using FFT
figure;

for i = 1:length(frequencies)
    subplot(length(frequencies), 1, i);
    signal_fft = fft(sine_waves(i, :));
    signal_fft_magnitude = abs(signal_fft);
    frequencies_fft = sampling_rate * (0:(length(time)/2))/length(time);
    plot(frequencies_fft, signal_fft_magnitude(1:length(time)/2+1));
```

```
    title(['Frequency Recovery (Original Frequency: '
num2str(frequencies(i)) ' Hz)']);
    xlabel('Frequency (Hz)');
    ylabel('Magnitude');
end
```