# Assignment 3: Learning and Memory PSY 306 (Winter 2024)

**Name: Vansh**

**Roll Number: 2021363**

---

**PART B**

**1.**

**5 participants performed a task where they were presented with a list of words on a computer screen. The words were presented one at a time. There were a total of 16 words in each list. After the presentation of each word in one list, the participants were asked to recall that list. The given Excel file contains the data of the words recalled from each list, and each sheet contains data for a total of 198 trials per participant.**
**Each row in the 'LM_A3_Data1.xlsx' file represents one trial.**
**Each column represents the serial position of the recalled item/word. The data file contains a numerical identifier for each response made by the participant during the free recall period.**
**The integers in the data file correspond to the serial position of the recalled item. For example, integer 15 will correspond to the word which was presented at the 15th position in the list of 16 words.**
**-1 represents intrusions ( word repeated from the previous list)**
**0 represents no item recalled at that position**

**Now do the following**

**A) Count the frequency of each item/word recalled by each participant. Plot a simple line and marker plot for each participant. Create a larger plot with 5 subplots, each containing the data for one participant. Identify and explain the effect associated with memory recall that can be observed from the data.**

**[3+1 Points]**

**Hint: Create a larger figure with five subplots (positioned as 5 rows x 1 columns); Indicate the participant number on top of each subplot as the title.**

**Python Code:**

```
from openpyxl import load_workbook
import matplotlib.pyplot as pyplt

# Load Excel file
excel_file = 'LM_A3_Data1.xlsx'

# Open the workbook
workbook = load_workbook(excel_file)

# Initialize a list to store the content of each sheet
content_list = []
```

```
# Iterate over each sheet and store its content into a list
for sheet_name in workbook.sheetnames:
    sheet = workbook[sheet_name]
    sheet_content = []

    # Iterate over rows in the sheet
    for row in sheet.iter_rows(values_only=True):
        # Append each row as a list to the sheet_content
        sheet_content.append(list(row))

    # Append the content of the sheet to the main list
    content_list.append(sheet_content)

word_freq={-1:0, 0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0,
13:0, 14:0, 15:0, 16:0}

data_participant_1=[]
data_participant_2=[]
data_participant_3=[]
data_participant_4=[]
data_participant_5=[]


for t1 in range (0,198) :
    for t2 in range (0,16) :
        word_freq[content_list[0][t1][t2]]+=1
for t1 in range (1,17) :
    data_participant_1.append(word_freq[t1])
    word_freq[t1]=0

for t1 in range (0,198) :
    for t2 in range (0,16) :
        word_freq[content_list[1][t1][t2]]+=1
for t1 in range (1,17) :
    data_participant_2.append(word_freq[t1])
    word_freq[t1]=0

for t1 in range (0,198) :
    for t2 in range (0,16) :
        word_freq[content_list[2][t1][t2]]+=1
for t1 in range (1,17) :
    data_participant_3.append(word_freq[t1])
    word_freq[t1]=0
```

```
for t1 in range (0,198) :
    for t2 in range (0,16) :
        word_freq[content_list[3][t1][t2]]+=1
for t1 in range (1,17) :
    data_participant_4.append(word_freq[t1])
    word_freq[t1]=0

for t1 in range (0,198) :
    for t2 in range (0,16) :
        word_freq[content_list[4][t1][t2]]+=1
for t1 in range (1,17) :
    data_participant_5.append(word_freq[t1])
    word_freq[t1]=0

print(data_participant_1)
print()
print()

print(data_participant_2)
print()
print()

print(data_participant_3)
print()
print()

print(data_participant_4)
print()
print()

print(data_participant_5)
print()
print()


# List of all data lists
data_lists = [
    data_participant_1, data_participant_2, data_participant_3,
    data_participant_4, data_participant_5
]

# Create a larger figure with five subplots (positioned as 5 rows x 1 column)
```

```
fig, line = pyplt.subplots(5, 1, figsize=(8, 12))

# Plot each list on a separate subplot
for i, data in enumerate(data_lists, start=1):
    # Generate x-axis values (1 to 16)
    x_values = range(1, len(data) + 1)
    # Plot the data on the subplot
    line[i-1].plot(x_values, data, marker='o', linestyle='-')
    # Set x-axis ticks to show every number from 1 to 16
    line[i-1].set_xticks(range(1, len(data) + 1))
    #set labels
    line[i-1].set_xlabel('Word Recalled')

    line[i-1].set_ylabel('Frequency')
    # Specifying titel
    line[i-1].set_title(f'Participant {i}')

# Add common labels
fig.suptitle('Simple Line and marker plot')

# Adjust layout
pyplt.tight_layout()

# Show the plot
pyplt.show()
```

**Graph:**

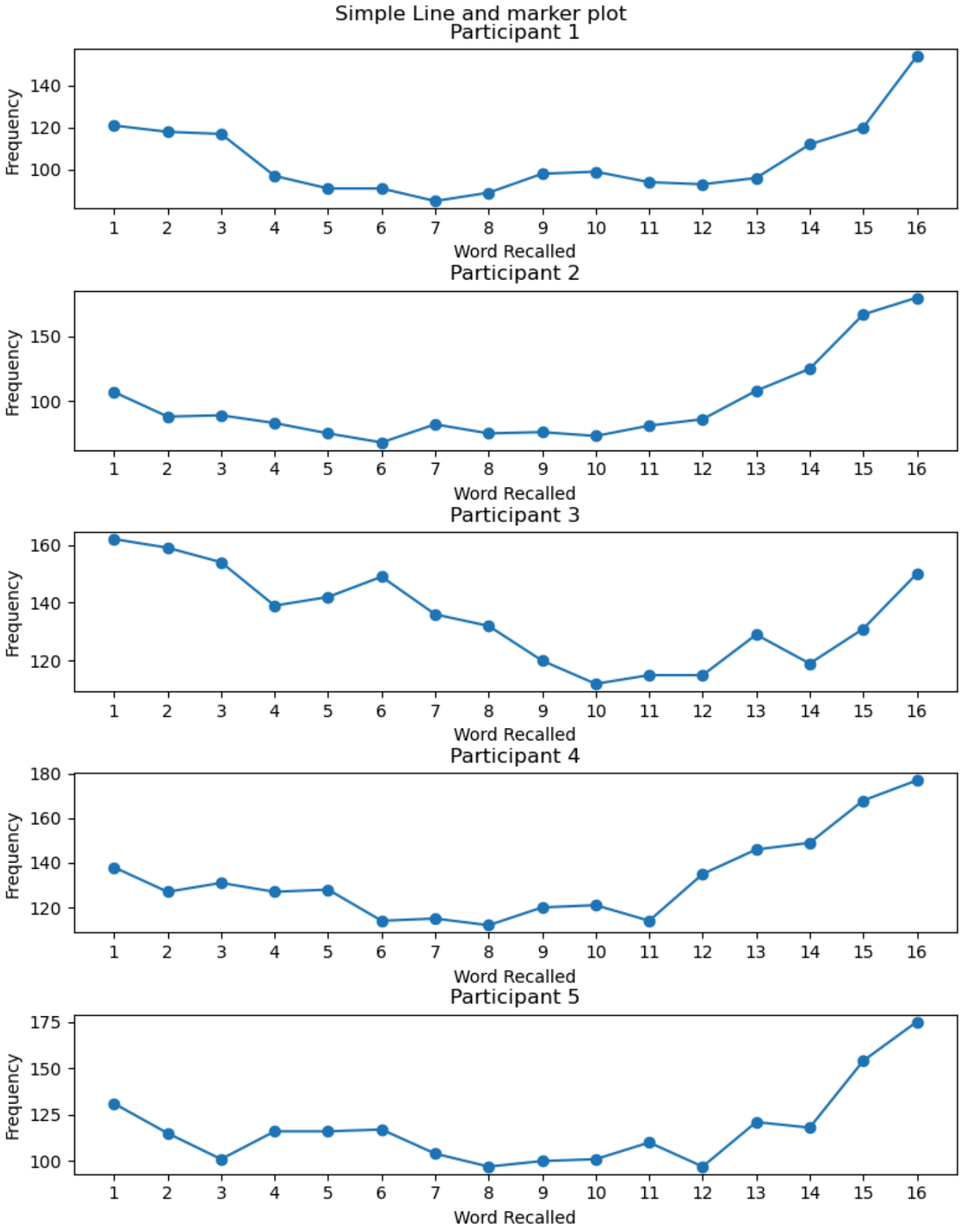

Simple Line and marker plot
Participant 1
Frequency
Word Recalled
Participant 2
Frequency
Word Recalled
Participant 3
Frequency
Word Recalled
Participant 4
Frequency
Word Recalled
Participant 5
Frequency
Word Recalled

**Figure Caption:**
The figure presented showcases a series of line and marker plots, each corresponding to one of five participants in a memory recall experiment. Each subplot represents the frequency of correctly recalled words at different serial positions from a list of 16 words during 198 trials. The frequency of recall is plotted on the y-axis and the serial positions are plotted along the x-axis. The line connecting markers on the plots illustrates the trend of recall for each word position for each participant. Significantly, the plots may suggest the presence of serial position effects, such as primary and recency, where words presented at the beginning and the end of the list are more frequently recalled than those in the centre. Omissions and Intrusions are coded as 0 and -1. This collective visualisation provides insight into the participants' memory performance patterns across repeated trials.

**Explanation:**

1. The effect that can be observed from the data is the “serial position effect,” The serial position effect is characterised by the tendency of a person to recall the first (primacy effect) and last (recency effect) items in a sequence more easily than the middle items.

2. The primacy effect is indicated by the higher frequency of words being recalled from the initial positions of the list. This occurs because the first few items are more effectively transferred to long-term memory due to the increased time to encode and consolidate them before additional information is processed.

3. The recency effect is shown by the uptick in frequency of the words recalled at the final positions of the list. This is attributed to the short-term memory's recency bias, where the most recently presented items are still available in working memory and thus are more readily retrieved.

4. These effects combined create a U-shaped curve when plotting recall performance across the serial positions.The middle positions typically show a lower frequency of recall, supporting the idea that items present in the beginning and at the end are more likely to be recalled than those at centre.

5. Each participant's plot demonstrates variations in the strength of the primacy and recency effects, which may be due to individual differences in memory processing or strategies employed during recall.

**B) For each participant, compute the distance from the last item recalled within each trial, excluding zeros and -1, by subtracting the item at the current index (ith) from the subsequent item at the adjacent index (ith+1). Count the frequency of each value and create a larger plot with five histograms (as subplots), each displaying the data for one participant.**
**What information can be drawn from the distribution (of the plot) with respect to the distance from the last recalled item?**

**[4+2 Points]**

**Hint: Create a larger figure with five subplots (positioned as 5 rows x 1 columns);**
**Indicate the participant number on top of each subplot as the title.**

**Python Code:**

```
from openpyxl import load_workbook
import matplotlib.pyplot as pyplt


# Load Excel file
excel_file = 'LM_A3_Data1.xlsx'
```

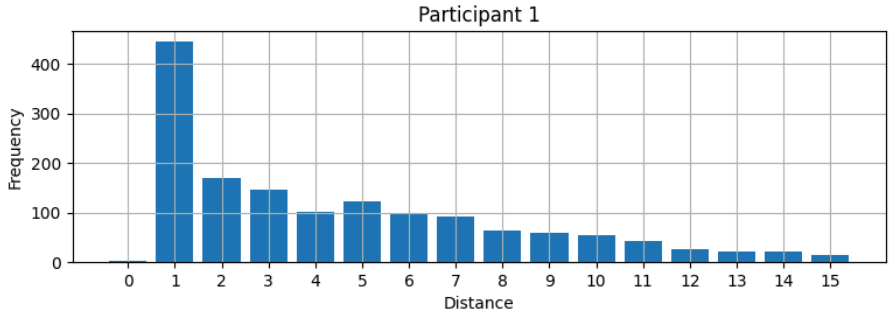
```
# Open the workbook
workbook = load_workbook(excel_file)

# Initialize a list to store the content of each sheet
content_list = []

# Iterate over each sheet and store its content into a list
for n_sheet in workbook.sheetnames:
    file_s = workbook[n_sheet]
    cont_s = []

    # Iterate over rows in the sheet
    for row in file_s.iter_rows(values_only=True):
        # Append each row as a list to the sheet_content
        cont_s.append(list(row))

    # Append the content of the sheet to the main list
    content_list.append(cont_s)

list_1=[0]*16
list_2=[0]*16
list_3=[0]*16
list_4=[0]*16
list_5=[0]*16

front=0
end=0

for i in range (0,198) :
    for k in content_list[0][i] :
        if (k==0 or k==-1) :
            pass
        else :
            if (front<=0) :
                front=k
            else :
                if (front>k) :
                    #if condition
                    list_1[front-k]+=1
                else :
                    #else condition
                    list_1[k-front]+=1
                front=k
```

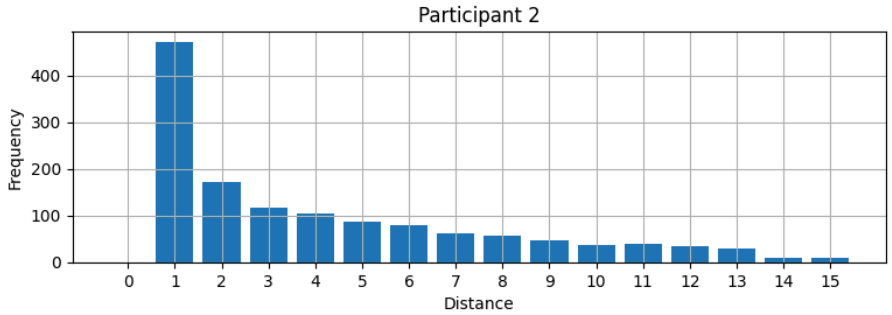
```
    front=0

for i in range (0,198) :
    for k in content_list[1][i] :
        if (k==0 or k==-1) :
            pass
        else :
            if (front<=0) :
                front=k
            else :
                if (front>k) :
                    list_2[front-k]+=1
                else :
                    list_2[k-front]+=1
                front=k
    front=0

for i in range (0,198) :
    for k in content_list[2][i] :
        if (k==0 or k==-1) :
            pass
        else :
            if (front<=0) :
                front=k
            else :
                if (front>k) :
                    list_3[front-k]+=1
                else :
                    list_3[k-front]+=1
                front=k
    front=0

for i in range (0,198) :
    for k in content_list[3][i] :
        if (k==0 or k==-1) :
            pass
        else :
            if (front<=0) :
                front=k
            else :
                if (front>k) :
                    list_4[front-k]+=1
                else :
```

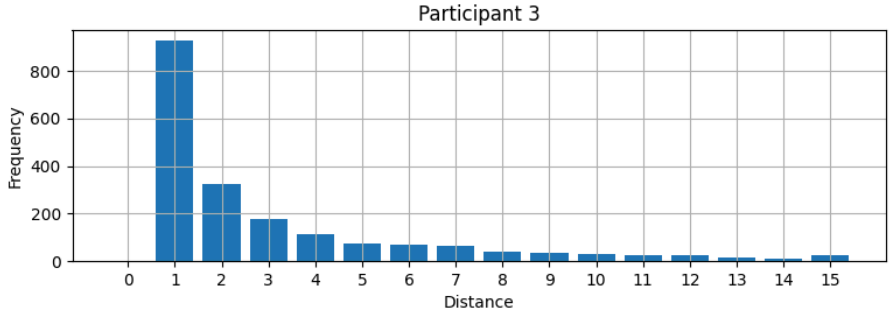
```
                    list_4[k-front]+=1
                front=k
    front=0

for i in range (0,198) :
    for k in content_list[4][i] :
        if (k==0 or k==-1) :
            pass
        else :
            if (front<=0) :
                front=k
            else :
                if (front>k) :
                    list_5[front-k]+=1
                else :
                    list_5[k-front]+=1
                front=k
    front=0

# Create a list of these lists for easier iteration
data_lists = [list_1, list_2, list_3, list_4, list_5]

# Creating the figure and subplots
plot, line = pyplt.subplots(5, 1, figsize=(10, 15))

# Plotting each list in its subplot
for a, j in enumerate(line):
    j.bar(range(16), data_lists[a])
    j.set_title(f'Participant {a+1}')
    j.set_xticks(range(16))
    #setting y label
    j.set_ylabel('Frequency')
    #setting x label
    j.set_xlabel('Distance')


# Add common labels
plot.suptitle('Histograms')
pyplt.subplots_adjust(hspace=0.5)
pyplt.show()
```

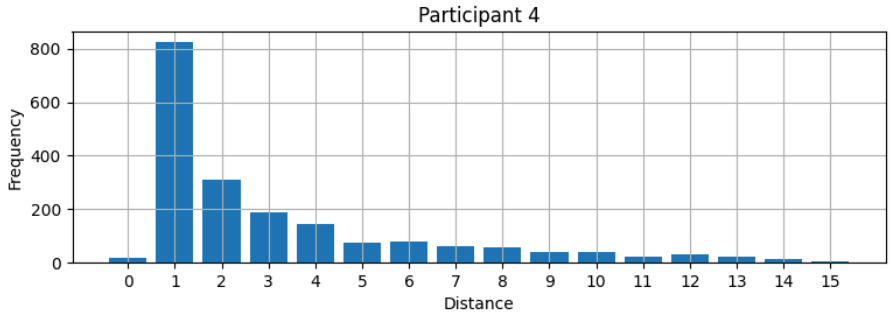
**Graph:**

Histograms


Participant 1
Frequency
Distance


Participant 2
Frequency
Distance


Participant 3
Frequency
Distance


Participant 4
Frequency
Distance

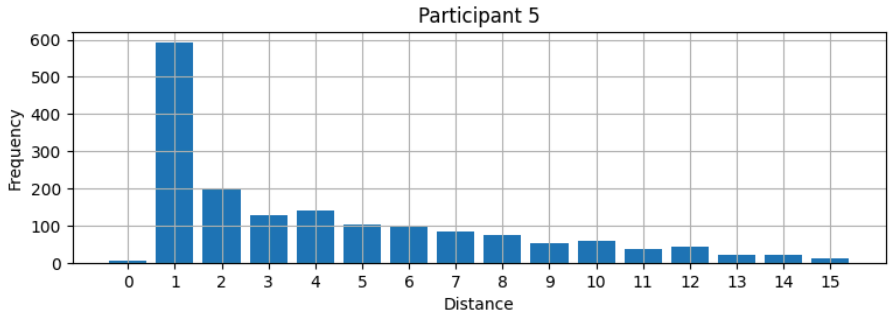

Participant 5
Frequency
Distance

**Figure Caption:** The figure depicts five histograms, each representing the frequency distribution of the distances between consecutively recalled items for five different participants in a memory recall task. The x-axis indicates the distance between the positions of consecutively recalled items from the original list, excluding repetitions and non-recalled items. The y-axis shows the frequency of these distances occurring across 198 trials. A clear trend is visible across all participants: items with an adjacent serial position (distance of 1) are most frequently recalled in sequence, suggesting a strong associative chaining or serial chaining effect in memory recall. As the distance increases, the frequency of recall significantly diminishes, indicating a lesser likelihood of long jumps in the recall sequence. The distributions vary slightly between participants, which may reflect individual differences in memory recall strategies or associative strengths.

**Insights:**

1. Preference for Adjacent Recall: There's a high frequency of '1's across all participants, which indicates that items next to each other in the list are often recalled in sequence. This suggests a strong chaining effect, where recalling one item serves as a cue for the next item in the list.
2. Decreasing Frequency with Increasing Distance: For all participants, as the distance increases, the frequency of that distance occurring tends to decrease. This suggests that participants are less likely to jump large distances between the serial positions of the list when recalling items, which indicates that items closer in the list are more likely to be associated or chained together in memory.
3. Low Frequency of Long Distance Recalls: There are very few instances of recalls with long distances between them, especially distances greater than 5. This may indicate that participants generally do not recall items in a random order but rather follow the sequence of the list to some extent.

**2.**

**Use the data given in LM_A3_Data2.xls | An experimenter recorded and pre-processed EEG data from 20 participants on an auditory oddball task playing them standard and deviant tones. The interstimulus intervals between the two tones were manipulated at levels – 0.75 s, 1.5s, 3s, 8s, and 9s as the EEG traces evoked by both standard and deviant tones were measured (1000 Hz sampling rate) from the participants' brains. Each Excel file sheet has data for both standard tone (beginning from cell 'B3') and deviant tone (beginning from cell 'B25'). For each of the above tones there is a 20 (participants) x 100 (time point) matrix in each sheet. Do the following…**

**[All figures/schematics should be properly labelled and should have accompanying captions/legends to provide all information necessary to interpret the same…]**

**A) Make a figure with five subplots – one for each interstimulus interval. In each subplot, graph the average EEG response (across 20 participants) from standard and deviant tones in blue and red, respectively.**

**[4 points]**

**Python Code:**

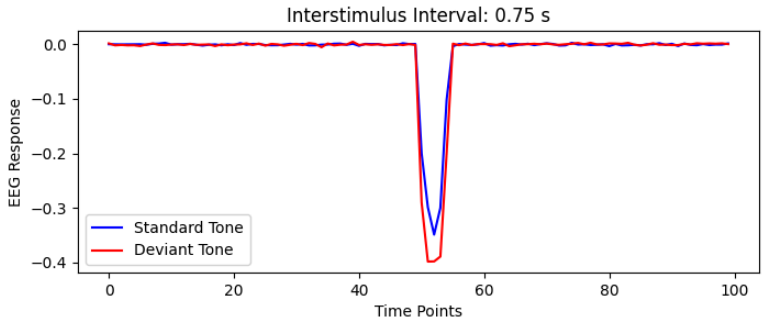
```
import xlrd
import matplotlib.pyplot as plt

def extract_data_from_xls(file_path):
    # Open the workbook
    workbook = xlrd.open_workbook(file_path)

    # List to store extracted data
```

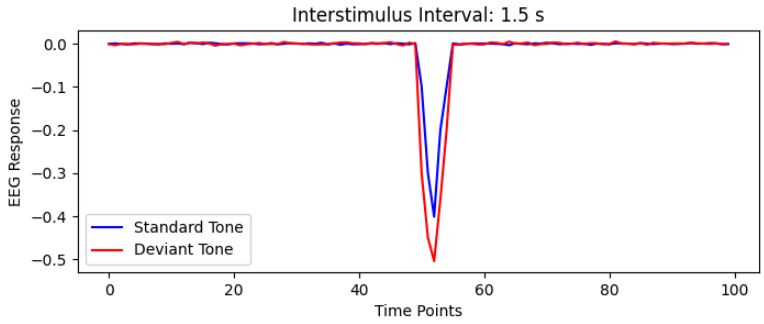
```
    all_data = []

    # Iterate through each sheet
    for sheet in workbook.sheets():
        sheet_data = []
        # Iterate through each portion (two portions as specified)
        for row_start in [2, 24]:  # Rows are 0-indexed, so 2 is B3, 24 is B25
            row_end = row_start + 19  # 20 rows per portion
            col_end = 99  # 100 columns (0-indexed)

            # Extract data from the current portion
            portion_data = []
            for row in range(row_start, row_end + 1):
                  row_values = sheet.row_values(row, start_colx=1, end_colx=col_end +
2)  # Start from B, so 1
                portion_data.append(row_values)

            sheet_data.append(portion_data)

        all_data.append(sheet_data)

    return all_data

# Usage example
file_path = "LM_A3_Data2.xls"
data = extract_data_from_xls(file_path)

standard=[]
deviant=[]



for t1 in range (0,5) :
    temp=[]
    for t2 in range (0,100) :
        temp1=0.0
        for k in range (0,20) :
            temp1+=data[t1][0][k][t2]
            temp1/=20
        temp.append(temp1)
    standard.append(temp)

for t1 in range (0,5) :
```

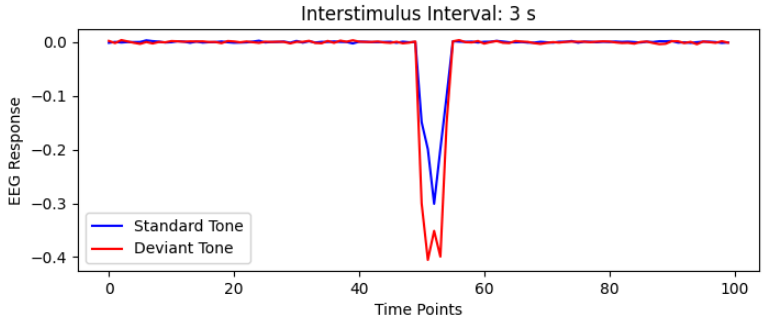
```
    temp=[]
    for t2 in range (0,100) :
        temp1=0.0
        for k in range (0,20) :
            temp1+=data[t1][1][k][t2]
            temp1/=20
        temp.append(temp1)
    deviant.append(temp)

# Create time points (assuming 100 time points)
time_points = range(100)


# Create a figure with subplots
fig, axs = plt.subplots(5, 1, figsize=(10, 15), sharex=True, sharey=True)


# Plot each interstimulus interval in a separate subplot
for i, (std_data, dev_data) in enumerate(zip(standard, deviant), start=1):

    ax = axs[i-1]  # Select the subplot for this interval
    ax.plot(time_points, std_data, label=f'Standard', color='blue', alpha=0.8)
    ax.plot(time_points, dev_data, label=f'Deviant', color='red', alpha=0.8)
    ax.set_title(f'Interstimulus Interval {i}')
    ax.legend()
    ax.grid(True)
    # Specifying labels
    axs[i-1].set_xlabel('Time Points')

    axs[i-1].set_ylabel('EEG Response')


# Labeling axes and showing the plot
fig.suptitle('Average EEG Response for Standard and Deviant Tones')
plt.subplots_adjust(hspace=0.5)
plt.show()
```

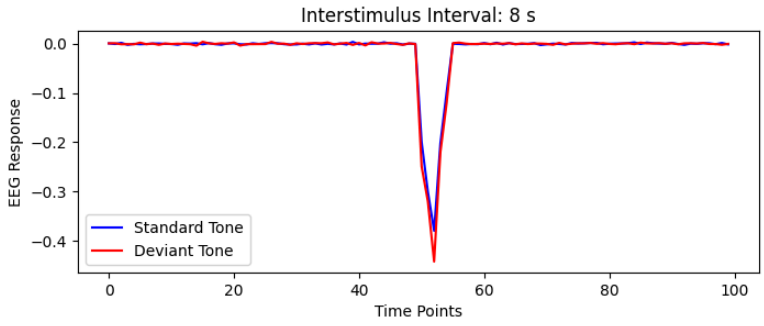
**Graph**

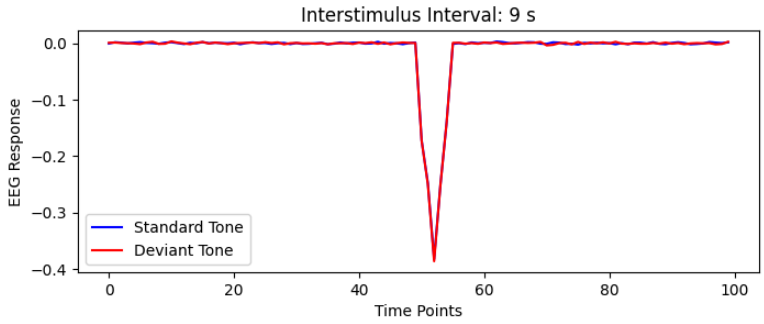
Average EEG Response for Standard and Deviant Tones




14 Assigr

**Figure caption:** Average EEG Responses to Standard and Deviant Tones at Various Interstimulus Intervals. This multi-panel graph displays the averaged electroencephalogram (EEG) responses from 20 participants to standard (blue) and deviant (red) auditory tones over a series of interstimulus intervals (ISIs). Each subplot corresponds to a different ISI, ranging from 0.75 seconds to 9 seconds, illustrating the brain's electrical activity in response to each tone type over 100 time points. It is observed that a diminishing distinction between the two tones in the participant's responses appear as the interstimulus interval increases. The V-shaped pattern observed in the subplots represents the brain's mechanism for identifying and categorising auditory stimuli.

**B) Analyze the data from each interstimulus interval statistically and report the time scale of echoic memory. Explain the cognitive science consistent rationale behind the calculation and reported time scale.**

**[4 + 2 points]**

**Hint: Carefully inspect the correctly created figure above for clues.**

**Python Code:**

```
import pandas as pnd
import numpy as nmp
from scipy import stats

#Path to the excel file
path = "LM_A3_Data2.xls"
interstimulus = [0.75, 1.5, 3, 8, 9]
q=0
temp1=5
temp=0

#Running the loop 5 times for every interstimulus interval
while(temp1) :
    q+=1
    #Extracting from excel file and storing in the variable dataframe
        dataframe  =  pnd.read_excel(path,  sheet_name=f"{interstimulus[5-temp1]}s",
header=None)
    data_dvn = dataframe.iloc[24:44, 1:101].to_numpy()
    q-=1
    data_std = dataframe.iloc[2:22, 1:101].to_numpy()
    #Calculating average standard and deviant EEG response
    dvn_average = nmp.mean(data_dvn, axis=0)
    q+=1
    std_average = nmp.mean(data_std, axis=0)
    #statistical test
    stat_t, value_p = stats.ttest_rel(std_average, dvn_average, axis=0)
    q-=1
```

```
    print(f"Interval: {interstimulus[5-temp1]}s, p-value of interstimulus interval:
{value_p}")
    if (not(temp!=0) and not(value_p<=0.05)):
        q+=1
        temp=1
        result=f"Time scale of echoic memory: {interstimulus[5-temp1]}s"
    temp1-=1
    q=0
#Printing result
print(result)
```

**Output on terminal:**

Interval: 0.75s, p-value of interstimulus interval: 0.036379484934067195
Interval: 1.5s, p-value of interstimulus interval: 0.03305741877282153
Interval: 3s, p-value of interstimulus interval: 0.04053897882280644
Interval: 8s, p-value of interstimulus interval: 0.06741863112782308
Interval: 9s, p-value of interstimulus interval: 0.3523953527790259
Time scale of echoic memory: 8s

**Cognitive science consistent rationale behind the calculation and reported time scale:**

Echoic memory is a form of sensory memory that temporarily holds sound information. To determine how long echoic memory lasts, we must identify the interstimulus interval at which the brain fails to differentiate between standard and deviant tones.
For this purpose, I have conducted paired t-tests at each time point across various interstimulus intervals. The point at which the brain can no longer detect differences between the two tones is identified when the p-value surpasses 0.05, occurring at approximately 8 seconds.
This method is based on cognitive science principles that seek to explore how the brain processes auditory signals. Echoic memory, which has a limited capacity and duration, is key in this process. By identifying the interval at which the ability to distinguish sounds diminishes, we can gauge the duration of echoic memory.