Assignment 2: Learning and Memory PSY 306 (Winter 2023)

Name: Utkarsh Arora

Roll Number: 2020143

Instructions: Please write your own responses and do not copy or lift text/code from any source. If you are referring to credible external sources other than the attached paper for your answers, please cite those sources (within the body of text and the provide a reference list at the end) in the APA citation format (https://www.mendeley.com/guides/apa-citation-guide). Word limits given are indicative and less than the indicated numbers may also be used.

Please download this MS word question-cum-response template to TYPE your answers and feel free to add sheets as required. Convert this document to a PDF before submitting. Please note that answers in this template only will be evaluated and hand-written or scanned answer sheets will not be evaluated. Please submit ONLY ONE PDF and no extra files as it increases the time to evaluate them. DO NOT change the basic structure of the template. DO NOT remove the marks assigned for each question.

[Strict deadline for submission: 23 March 2023, 11 PM]

Q2) Please read the following for this question:

- **A researcher recorded electromyogram (EMG) from the extraocular muscles of a human participant as a tone was delivered through headphones and air-puff delivered to the eyes through an apparatus to the participant. The tone stimulus onset is at time = 0 ms (beginning of the trial) and continues until 650 ms. The air-puff stimulus onset is at time = 600 ms and continues for the next 50 ms.**
- **The above was done for five trials/day for four subsequent days and the EMG responses recorded as data. Download the attached data file- Data-Assignment2A.xlsx**
- **Each sheet of the excel file contains EMG recording from one day of experiment. Each sheet has 5 rows (trials) x 1000 columns (EMG amplitudes recorded at an interval of 1 millisecond). Thus each row has 1000 ms (1 second) of recording.**

  Now do the following…

Insert a figure (wherever required) and paste the MATLAB/Python code for the same. All figures must be properly labelled, carry necessary units of measurement with accompanying captions/legends to provide all information necessary to interpret the figures.

A) Run the following steps…

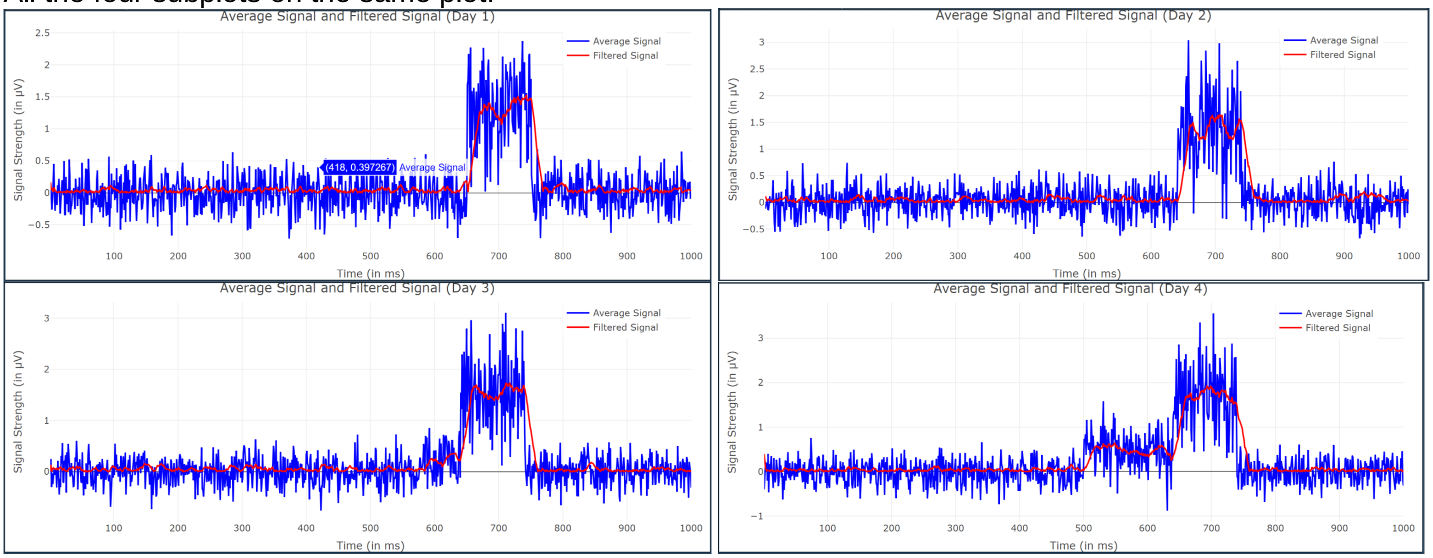
- **Take the average of data across all trials per day for each time point to get one averaged signal per day.**
- **Run a 'moving average filter' across the averaged signal with a window width of 20 ms to get a filtered signal. Ensure that the raw and filtered signal are of the same length.**
- **Do a full wave rectification of the above moving average filtered signal.**
- **Plot the amplitude vs time of the raw signal (as blue curve) - one signal for each day in four different subplots of one bigger plot.**
- **Plot the amplitude vs time of the filtered and rectified signal (as red curve) – one signal for each day on top of the raw signal in the same subplots.**

After creating the above plot, explain the learning mechanism evident in the above plot with all necessary components of learning that you think are involved in this case. Calculate exact time points of the peaks from the above data to draw your quantitative conclusions about the learning mechanism and its components.

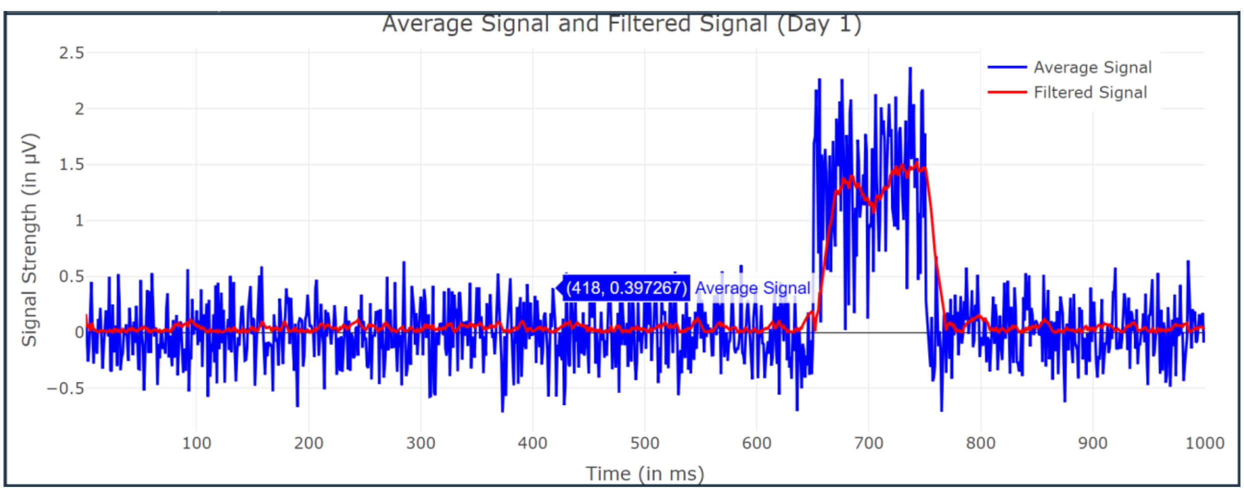
[10 points]

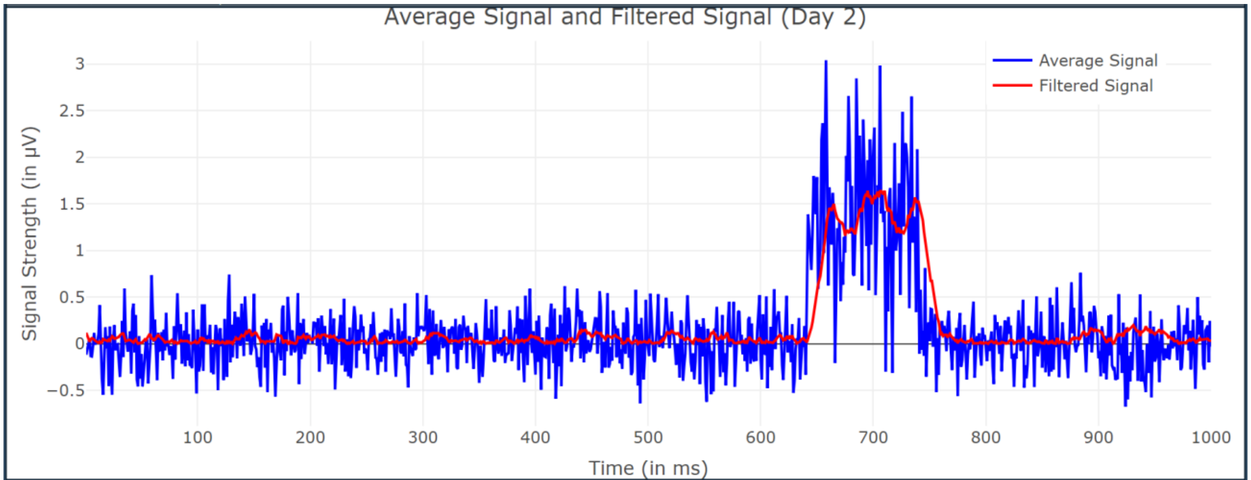
I have first taken the average of the data across all trials per day for each time point to get one averaged signal per day. Then I have taken a rolling average filter across the averaged data with a window of 20 miliseconds to get the filtered signal. Then, I have done a full wave rectification of the moving average filtered signal. Finally, I have plotted the amplitude vs time of the raw signal and the amplitude vs time of the filtered signal on the same plot.

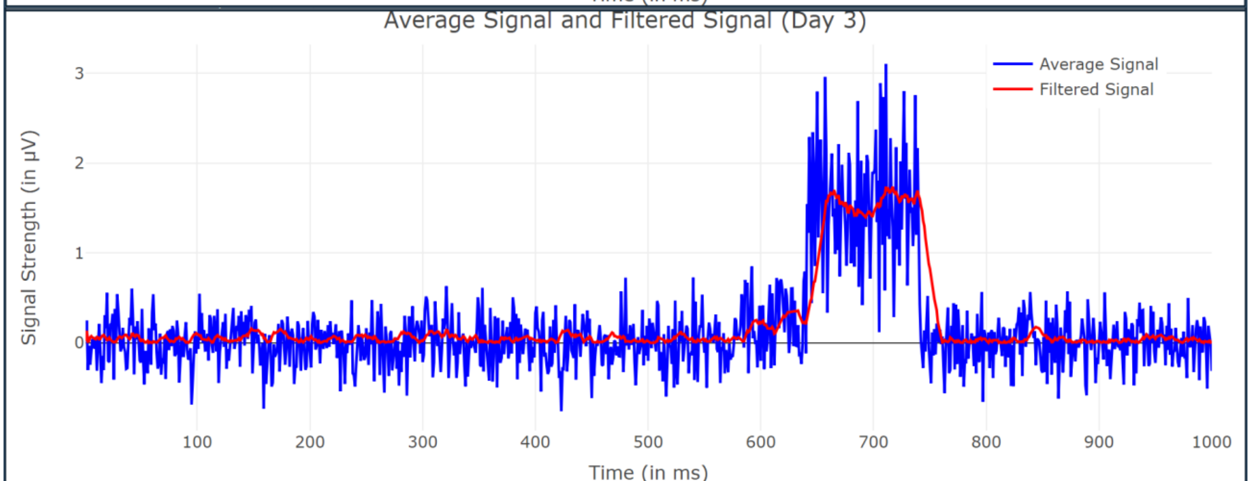
All the four subplots on the same plot:

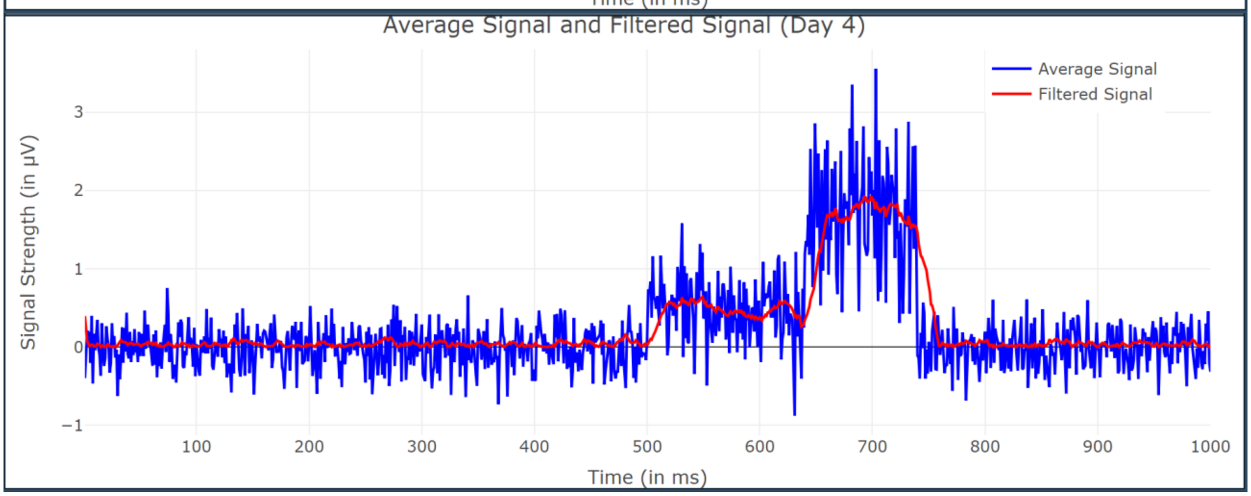


Individual plots:









Absolute peak and timings:
Day 1: 1.530592 μV at 743 ms
Day 2: 1.641685 μV at 710 ms
Day 3: 1.736882 μV at 717 ms
Day 4: 1.92855 μV at 699 ms

From these plots, it is observable that there is a clear increase in the amplitude of the EMG signal in response to the tone stimulus over the four days of the experiment. There is also a decrease in the time at which the amplitude starts to show peaking characteristics. There is also a decrease in time at which the peak is

achieved. This suggests that the participant is learning to anticipate the air-puff stimulus based on the tone stimulus.
The increased amplitude of the EMG signal would indicate the activity of extraocular muscles has increased, which would suggest that the participant has been preparing to blink in response to anticipated air-puff stimulus. This implies that Classical or Pavlovian conditioning could be the possible learning mechanism involved here.
In classical conditioning, the neutral stimulus does not initially evoke a response in classical conditioning, but after being repeatedly paired with the unconditioned stimulus, it eventually starts to elicit the same response as the unconditioned stimulus. Here, we have paired **the tone**, an initially neutral stimulus, with **the airpuff**, an unconditioned stimulus, which elicits **the eyeblink**, an unconditioned response. With time, the neutral stimulus eventually develops into a conditioned stimulus that triggers a conditioned reaction (the eye-blink).
To measure the learning mechanism, we can calculate the time points of the peaks in the rectified and filtered signals, which correspond to the maximum amplitude of the EMG response. We can use these time points to estimate the peak latency of the learned response. In the code, we have calculated the exact time points and observed that the peak times start to reach earlier as we progress through the days. This suggests that the participant is preparing to blink earlier in response to the stimuli.
The observed increase in EMG amplitude in response to the tone stimulus suggests that the participant became more sensitive to the stimuli over time, which may reflect improvements in attention and perceptual processing. Furthermore, the decrease in times to peak values suggests that the participant became more efficient in preparing and executing the appropriate eye movements in response to the stimuli, which may reflect improvements in motor planning and feedback processes. These findings indicate that the participant has successfully learned to anticipate the air-puff stimulus based on the tone stimulus through classical conditioning.

R code:

```
library(ggplot2)
library(readxl)
library(plotly)
library(gridExtra)

#Taking the average of data across all trials per day for each time point to get one averaged signal
#per day

df_day1 <- read.csv("C://Users//Utkarsh//Documents//homework//Learning and Memory//Assignment 2//day1.csv", header=FALSE)
df_day2 <- read.csv("C://Users//Utkarsh//Documents//homework//Learning and Memory//Assignment 2//day2.csv", header=FALSE)
df_day3 <- read.csv("C://Users//Utkarsh//Documents//homework//Learning and Memory//Assignment 2//day3.csv", header=FALSE)
df_day4 <- read.csv("C://Users//Utkarsh//Documents//homework//Learning and Memory//Assignment 2//day4.csv", header=FALSE)

avg_signal_day1 <- c()
avg_signal_day2 <- c()
avg_signal_day3 <- c()
avg_signal_day4 <- c()
```

```
for (i in df_day1){
  curr = mean(as.numeric(i))
  if (!is.na(curr)){
    avg_signal_day1 = append(avg_signal_day1, curr)
  }
}

for (i in df_day2){
  curr = mean(as.numeric(i))
  if (!is.na(curr)){
    avg_signal_day2 = append(avg_signal_day2, curr)
  }
}

for (i in df_day3){
  curr = mean(as.numeric(i))
  if (!is.na(curr)){
    avg_signal_day3 = append(avg_signal_day3, curr)
  }
}

for (i in df_day4){
  curr = mean(as.numeric(i))
  if (!is.na(curr)){
    avg_signal_day4 = append(avg_signal_day4, curr)
  }
}

#Calculating the rolling average

filter_day1 <- c()
filter_day2 <- c()
filter_day3 <- c()
filter_day4 <- c()

curr_full_1 = 0
curr_full_2 = 0
curr_full_3 = 0
curr_full_4 = 0

t <- c()


for (i in 1:1000){
  curr_day1 = avg_signal_day1[i]
  curr_day2 = avg_signal_day2[i]
  curr_day3 = avg_signal_day3[i]
  curr_day4 = avg_signal_day4[i]
```

```
  curr_full_1 = curr_full_1 + curr_day1
  curr_full_2 = curr_full_2 + curr_day2
  curr_full_3 = curr_full_3 + curr_day3
  curr_full_4 = curr_full_4 + curr_day4

  if (i<21){
    x = i
  }
  else{
    x = 20
    curr_full_1 = curr_full_1 - avg_signal_day1[i-20]
    curr_full_2 = curr_full_2 - avg_signal_day2[i-20]
    curr_full_3 = curr_full_3 - avg_signal_day3[i-20]
    curr_full_4 = curr_full_4 - avg_signal_day4[i-20]
  }

  filter_day1 = append(filter_day1, curr_full_1/x)
  filter_day2 = append(filter_day2, curr_full_2/x)
  filter_day3 = append(filter_day3, curr_full_3/x)
  filter_day4 = append(filter_day4, curr_full_4/x)
  t = append(t, i)
}

#Performing full wave rectification

for (i in 1:1000){
  filter_day1 = abs(filter_day1)
  filter_day2 = abs(filter_day2)
  filter_day3 = abs(filter_day3)
  filter_day4 = abs(filter_day4)
}


# Create four separate plots
p1 <- plot_ly() %>%
  add_lines(x = ~t, y = ~avg_signal_day1, line = list(color = "blue"), name = "Average
Signal") %>%
  add_lines(x = ~t, y = ~filter_day1, line = list(color = "red"), name = "Filtered Signal")
%>%
  layout(title = "Average Signal and Filtered Signal (Day 1)",
         xaxis = list(title = "Time (in ms)"),
         yaxis = list(title = "Signal Strength (in µV)"),
         showlegend = TRUE,
         legend = list(title = "Signal Type",
                       x = 0.8, y = 1)) %>%
  config(displayModeBar = FALSE)
```

```
p2 <- plot_ly() %>%
  add_lines(x = ~t, y = ~avg_signal_day2, line = list(color = "blue"), name = "Average
Signal") %>%
  add_lines(x = ~t, y = ~filter_day2, line = list(color = "red"), name = "Filtered Signal")
%>%
  layout(title = "Average Signal and Filtered Signal (Day 2)",
         xaxis = list(title = "Time (in ms)"),
         yaxis = list(title = "Signal Strength (in µV)"),
         showlegend = TRUE,
         legend = list(title = "Signal Type",
                       x = 0.8, y = 1)) %>%
  config(displayModeBar = FALSE)

p3 <- plot_ly() %>%
  add_lines(x = ~t, y = ~avg_signal_day3, line = list(color = "blue"), name = "Average
Signal") %>%
  add_lines(x = ~t, y = ~filter_day3, line = list(color = "red"), name = "Filtered Signal")
%>%
  layout(title = "Average Signal and Filtered Signal (Day 3)",
         xaxis = list(title = "Time (in ms)"),
         yaxis = list(title = "Signal Strength (in µV)"),
         showlegend = TRUE,
         legend = list(title = "Signal Type",
                       x = 0.8, y = 1)) %>%
  config(displayModeBar = FALSE)

p4 <- plot_ly() %>%
  add_lines(x = ~t, y = ~avg_signal_day4, line = list(color = "blue"), name = "Average
Signal") %>%
  add_lines(x = ~t, y = ~filter_day4, line = list(color = "red"), name = "Filtered Signal")
%>%
  layout(title = "Average Signal and Filtered Signal (Day 4)",
         xaxis = list(title = "Time (in ms)"),
         yaxis = list(title = "Signal Strength (in µV)"),
         showlegend = TRUE,
         legend = list(title = "Signal Type",
                       x = 0.8, y = 1)) %>%
  config(displayModeBar = FALSE)

# Combine the plots using the gridExtra package
grid.arrange(p1, p2, p3, p4, ncol = 2)

# Calculate the peaks
peak_day1 <- max(filter_day1)
peak_day2 <- max(filter_day2)
peak_day3 <- max(filter_day3)
peak_day4 <- max(filter_day4)
```

```
peak_time_day1 <- which(filter_day1 == peak_day1)
peak_time_day2 <- which(filter_day2 == peak_day2)
peak_time_day3 <- which(filter_day3 == peak_day3)
peak_time_day4 <- which(filter_day4 == peak_day4)

peak_day1
peak_time_day1

peak_day2
peak_time_day2

peak_day3
peak_time_day3

peak_day4
peak_time_day4
```

B) An experimenter carries out three pilot experiments of 30 trials each in human subjects to study the relationship between time (# trials) and Associative learning between the exposure to sets of environmental stimuli (Conditioned and Unconditioned Stimuli). She collects and averages the data across equal number of subjects for each pilot experiment. This data is entered in the Data-Assignment2B.xlsx. Each row = 1 pilot experiment. Each column is the value/magnitude of the CR (arbitrary units). Now carry out the following...

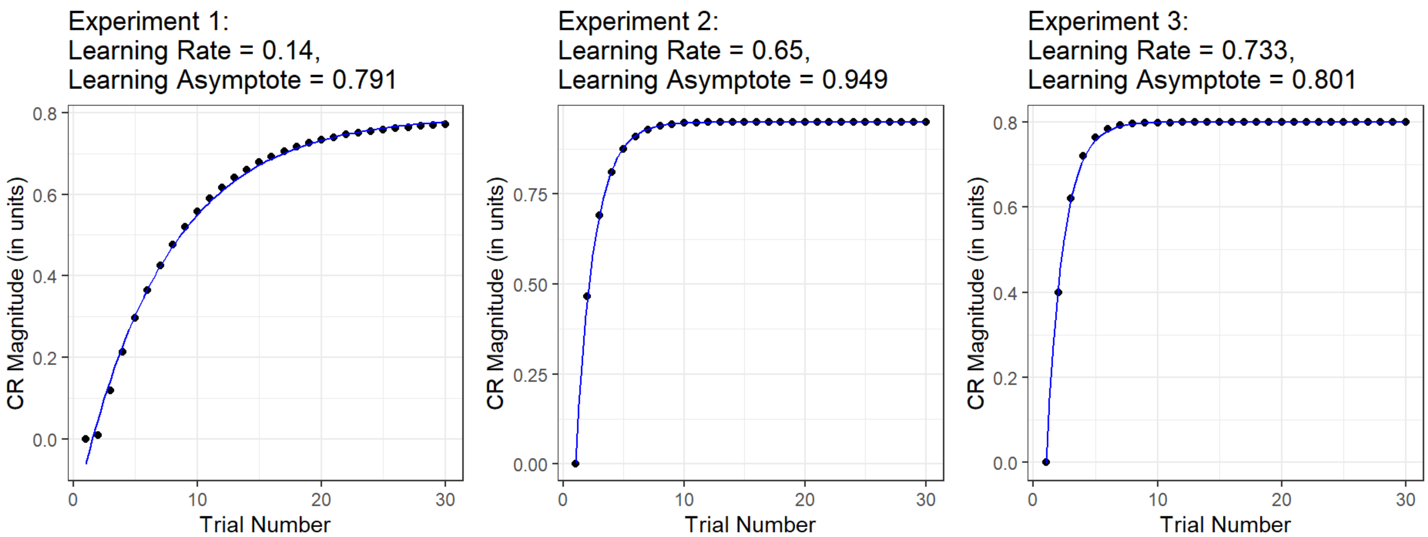
i) Computationally estimate the Rates and Asymptotes of Learning for the three pilot experiments. Create three subplots for three experiments as part of one larger plot to graph the individual data points (as open circle markers; black colour) and overlay of the learning curve (blue colour) on each subplot. Indicate the Learning rate and Learning asymptote on top of each subplot (as title).

Also, report any one metric of "goodness of fit" for each of the three learning curves to the underlying experimental data and briefly explain the quality of your curve fit to the experimental data based on the metric.

Hint: - Use unconstrained nonlinear optimization to find the optimal parameters of the negatively accelerated learning curve which best describes the relationship within the data, quantitatively.
For a measure of goodness of curve fit to the experimental data, explore and report any one of these metrics - sum of squared errors OR R square OR adjusted R square.

ii) Based on your analysis of the data what can you conclude about the intensities of the Unconditioned Stimuli in the three pilot experiments and why?

[8+2 points]



Goodness of Fit (Sum of Squared Errors):
Experiment 1: 0.007118844
Experiment 2: 0.0001884192
Experiment 3: 0.000441225

The Sum of Squared Errors (SSE) is a measure of how well the fitted curve matches the experimental data points. A lower SSE indicates a better fit.

In this case, the SSE values for Experiment 1, 2, and 3 are 0.007118844, 0.0001884192, and 0.000441225, respectively. These values suggest that the model fits the data very well for Experiments 2 and 3, as their SSE values are very low. For Experiment 1, the SSE value is higher, indicating that the model does not fit the data as well compared to the other experiments.

Overall, the curve fits to the experimental data are of high quality for Experiments 2 and 3, while it is still good but slightly less accurate for Experiment 1.

On analyzing the data, we can see that the intensity of the Unconditioned Stimuli (UCS) was different across the three pilot experiments.
Experiment 1 had a lower learning rate (0.14) and a lower asymptote (0.791) compared to Experiment 2, which had a higher learning rate (0.65) and a higher asymptote (0.949). Experiment 3 had a higher learning rate (0.733) compared to Experiment 1, but a lower asymptote (0.801) compared to Experiment 2.
According to the higher learning rate and asymptote, the intensity of the UCS was greatest in experiment 2. Experiment 1, on the other hand, had a lower intensity of UCS compared to Experiment 2, as shown by the lower asymptote, but the learning rate was also lower, implying a slower learning rate.
Experiment 3, meanwhile, had a higher learning rate than Experiment 1, but a lower asymptote than Experiment 2, which would indicate an intermediate intensity of the UCS.

Overall, these findings suggest that the intensity of the UCS plays an important role in the strength of the association between the UCS and CS, with higher intensity leading to stronger associations. However, other factors such as the timing and frequency of the UCS and CS presentations may also influence the strength of the association.

R code:

```
# Load the necessary packages
library(dplyr)
library(ggplot2)
```

```
library(minpack.lm)

# Load the data
df <- read.csv("C://Users//Utkarsh//Documents//homework//Learning and Memory//Assignment 
2//Data-Assignment2B.csv")
exp1 <- df$i..exp1
exp2 <- df$exp2
exp3 <- df$exp3
t <- df$trial_no

# Define the negatively accelerated learning curve model
neg_exp_func <- function(x, a, b, c) {
  a * (1 - exp(-b * x)) + c
}

# Fit the model to experiment 1
fit1 <- nlsLM(exp1 ~ neg_exp_func(t, a, b, c), start = list(a = 1, b = 0.1, c = 0.1))
b1 <- coef(fit1)['b']
a1 <- coef(fit1)['a']
c1 <- coef(fit1)['c']
asy1 <- a1 + c1

# Fit the model to experiment 2
fit2 <- nlsLM(exp2 ~ neg_exp_func(t, a, b, c), start = list(a = 1, b = 0.1, c = 0.1))
b2 <- coef(fit2)['b']
a2 <- coef(fit2)['a']
c2 <- coef(fit2)['c']
asy2 <- a2 + c2

# Fit the model to experiment 3
fit3 <- nlsLM(exp3 ~ neg_exp_func(t, a, b, c), start = list(a = 1, b = 0.1, c = 0.1))
b3 <- coef(fit3)['b']
a3 <- coef(fit3)['a']
c3 <- coef(fit3)['c']
asy3 <- a3 + c3

# Print the learning rate and asymptote for each experiment
cat(sprintf("Experiment 1: Learning Rate = %.3f, Learning Asymptote = %.3f\n", b1, a1))
cat(sprintf("Experiment 2: Learning Rate = %.3f, Learning Asymptote = %.3f\n", b2, a2))
cat(sprintf("Experiment 3: Learning Rate = %.3f, Learning Asymptote = %.3f\n", b3, a3))

# Plot the data and the fitted curves for each experiment
plot1 <- ggplot(df, aes(x = trial_no)) +
  geom_point(aes(y = exp1), color = "black") +
  stat_function(fun = neg_exp_func, args = list(a = a1, b = b1, c = coef(fit1)['c']), color = 
"blue") +
  ggtitle(paste0("Experiment 1: \nLearning Rate = ", round(b1, 3), ", \nLearning Asymptote = 
", round(asy1, 3))) +
```

```
  labs(x = "Trial Number", y = "CR Magnitude (in units)") +
  theme_bw()

plot2 <- ggplot(df, aes(x = trial_no)) +
  geom_point(aes(y = exp2), color = "black") +
  stat_function(fun = neg_exp_func, args = list(a = a2, b = b2, c = coef(fit2)['c']), color =
"blue") +
  ggtitle(paste0("Experiment 2: \nLearning Rate = ", round(b2, 3), ", \nLearning Asymptote =
", round(asy2, 3))) +
  labs(x = "Trial Number", y = "CR Magnitude (in units)") +
  theme_bw()

plot3 <- ggplot(df, aes(x = trial_no)) +
  geom_point(aes(y = exp3), color = "black") +
  stat_function(fun = neg_exp_func, args = list(a = a3, b = b3, c = coef(fit3)['c']), color =
"blue") +
  ggtitle(paste0("Experiment 3: \nLearning Rate = ", round(b3, 3), ", \nLearning Asymptote =
", round(asy3, 3))) +
  labs(x = "Trial Number", y = "CR Magnitude (in units)") +
  theme_bw()

grid.arrange(plot1, plot2, plot3, ncol = 3)

sse1 <- sum(resid(fit1)^2)
sse2 <- sum(resid(fit2)^2)
sse3 <- sum(resid(fit3)^2)

sse1
sse2
sse3
```