IN THE NAME OF GOD

HW3:

Neuroscience, Learning, Memory, Cognition Course

Sharif University of Technology

SaraRezanezhad99101643

June2023

PART I: PYTHON Exercise

1) Exercise Outline:
2) The Steinmetz dataset is a collection of neural recordings obtained from the brains of mice during a visual behavior task. The dataset contains 39 Neuropixels recordings, each of which consists of 400-700 neurons from different regions of the mouse brain. The recordings were made while the mice performed a visual task that involved detecting moving gratings on a screen. The dataset includes information about the stimuli presented to the mice, as well as the behavioral responses of the mice. The goal of the dataset is to provide a resource for researchers to study neural activity and its relationship to behavior in the mouse brain.
3) Load the given dataset (This is not the full dataset; it includes recorded neurons from visual cortex)

   By just uploading the file in my google drive, then running the following code the mission's done! 😊

✓ What was the mouse's name? 🐭

```
mouse name: Lederberg
```

✓ Count number of right, NoGo, Left trials?

We define 3 variables for every kind of trial and the use the for loop to obtain the number of each trial

```
dt = dat['bin_size']  # binning at 10 ms
NT = dat['spks'].shape[-1] # number of trials
right = 0
Left = 0
nogo = 0
for x in response:
  if x == -1:
    right = right+1
  if x == 1:
    Left = Left+1
  if x == 0:
    nogo = nogo+1
print(right)
print(Left)
print(nogo)
print(response.size)
```

```
141
135
64
340
250
```

✓ Please separate data of different trial types as described below and raster plot the spike_time/neuron for right, NoGo, Left for a trial.

The code written by me in the exercise Python file is part of a spike data analysis, where the spike data is separated into different trial types (NoGo, Left, and Right). It then defines a function called plot_spikes that generates a spike plot for a specific trial type with a specific time window.

Code description:

- `NoGo_trial = dat['spks'][:, response == 0, :]`:

This line creates a new variable called NoGo_trial which contains the spks data from the dat dictionary for all neurons and trials where the response is equal to 0.

- `Left_trial = dat['spks'][:, response == 1, :]`:

This line creates a new variable called Left_trial which contains the spks data from the dat dictionary for all neurons and trials where the response is equal to 1.

- `Right_trial = dat['spks'][:, response == -1, :]`:

This line creates a new variable called Right_trial which contains the spks data from the dat dictionary for all neurons and trials where the response is equal to -1.

- `def plot_spikes(spike_data, title, trial_idx, time_idx_start)`:

This line defines a function called plot_spikes that takes 4 arguments: spike_data (the spike data for a specific trial type), title (the title of the plot), trial_idx (the index of the trial to plot), and time_idx_start (the starting index of the time window to plot).

- `time_window = spike_data[:, trial_idx, time_idx_start:time_idx_start+25]`:

This line selects the appropriate time window for a specific trial by slicing the spike_data along the time axis for the specified trial_idx. The time window is defined as a 3D array with dimensions of (neurons, trials, time).

- `spike_times = np.argwhere(time_window)`:

This line extracts the spike times from the time_window array by finding the indices of all non-zero elements and returns their corresponding indices as a tuple.

- `time_axis = np.arange(time_idx_start, time_idx_start+25) * dt`:

This line creates a time axis for the plot by generating a 1D array from time_idx_start to time_idx_start+25 with each point spaced dt seconds apart.

- `plt.figure()`:

This line creates a new matplotlib plot figure.

- `plt.plot(spike_times * dt, spike_times[:, 0], 'o', markersize=2)`:

This line plots each spike time as a dot on the plot, with the x-coordinate being the spike time multiplied by the dt parameter, and the y-coordinate being the neuron ID.

- `plt.xlabel('Time (s)')`:

This line sets the x-axis label to "Time (s)".

- `plt.ylabel('Neuron ID')`:

This line sets the y-axis label to "Neuron ID".

- `plt.title(title)`:

This line sets the plot title to the title argument passed to the function.

- `plt.xlim([time_axis[0], time_axis[-1]])`:

This line sets the limits for the x-axis to be the first and last element of the time_axis array.

- `plt.show()`:

This line displays the plot.

- `plot_spikes(NoGo_trial, 'No-Go trials', -25, 0)`:

This line generates a spike plot for the "NoGo" trial type with a time window starting from time index 0.

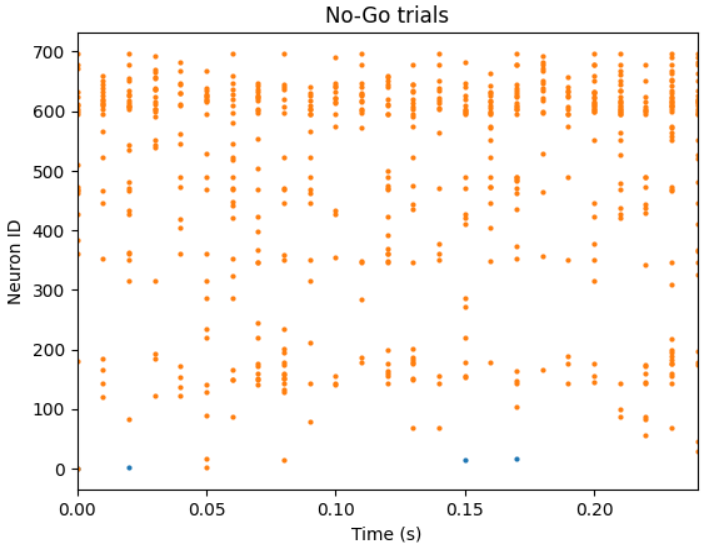
- `plot_spikes(Left_trial, 'Left trials', -25, 0)`:

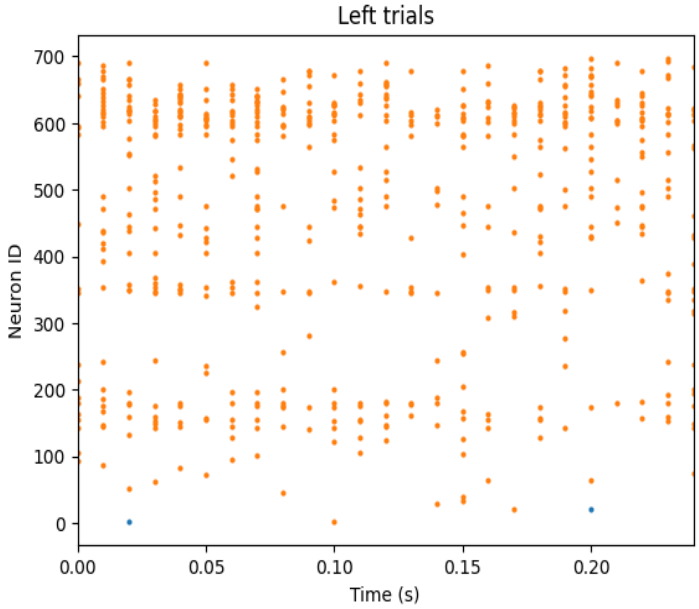
This line generates a spike plot for the "Left" trial type with a time window starting from time index 0.

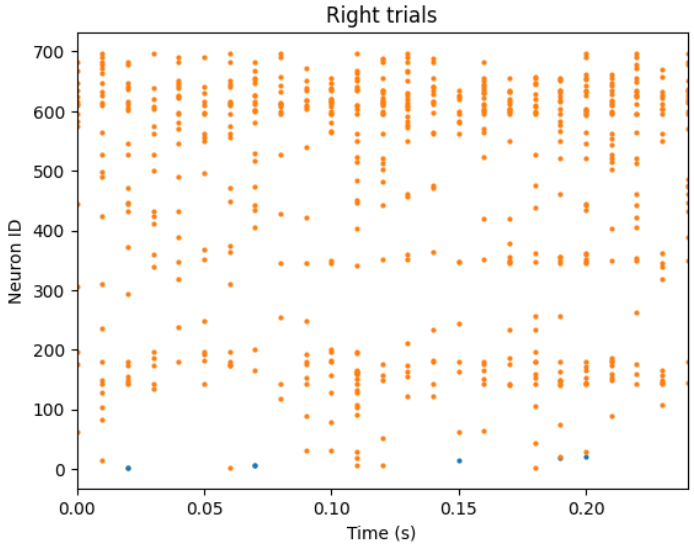
- `plot_spikes(Right_trial, 'Right trials', -25, 0)`:

This line generates a spike plot for the "Right" trial type with a time window starting from time index 0.

Figures:


No-Go trials
Neuron ID
Time (s)
0
100
200
300
400
500
600
700
0.00
0.05
0.10
0.15
0.20


Left trials
Neuron ID
Time (s)
0
100
200
300
400
500
600
700
0.00
0.05
0.10
0.15
0.20


Right trials
Neuron ID
Time (s)
0
100
200
300
400
500
600
700
0.00
0.05
0.10
0.15
0.20

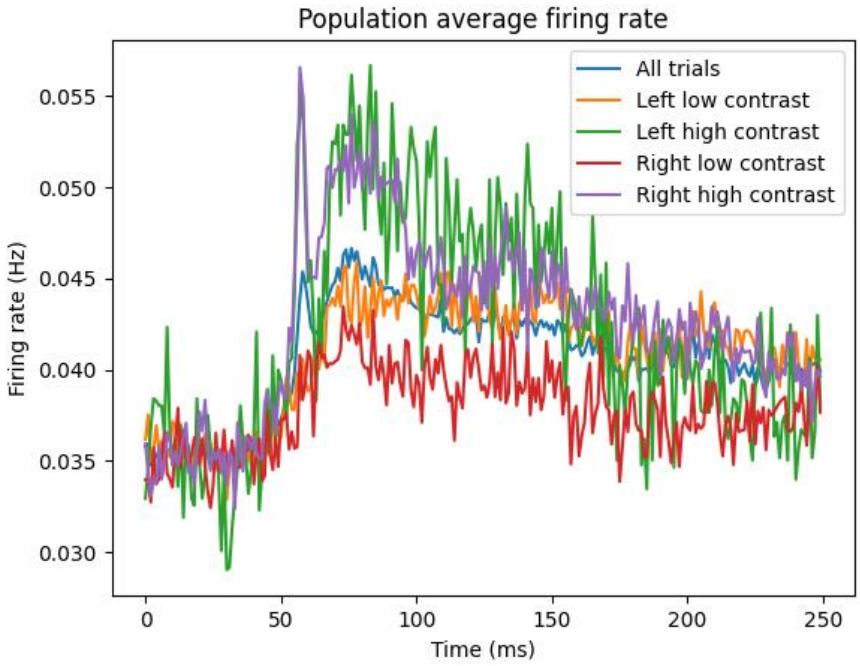
✓ Raster plot other trials, is there a similar pattern in same trial types? In which neurons? Which location? Describe it in your report.

4) Basic population average

```
response = dat['response']  # right - nogo - left (-1, 0, 1)
vis_right = dat['contrast_right']  # 0 - low - high
# Compute the average firing rate across all neurons and trials for each time point
avg_firing_rate = np.mean(dat['spks'], axis=(0, 1))  # shape: (T,)

# Separate the spike data for different stimulus types
low_contrast = (vis_right == 0)
high_contrast = (vis_right == 1)

# Compute the average firing rate for each combination of response and stimulus
left_low = np.mean(dat['spks'][:, (response == 1) & low_contrast, :], axis=(0, 1))  # shape: (T,)
left_high = np.mean(dat['spks'][:, (response == 1) & high_contrast, :], axis=(0, 1))  # shape: (T,)
right_low = np.mean(dat['spks'][:, (response == -1) & low_contrast, :], axis=(0, 1))  # shape: (T,)
right_high = np.mean(dat['spks'][:, (response == -1) & high_contrast, :], axis=(0, 1))  # shape: (T,)

# Plot the results
plt.figure()
plt.plot(avg_firing_rate, label='All trials')
plt.plot(left_low, label='Left low contrast')
plt.plot(left_high, label='Left high contrast')
plt.plot(right_low, label='Right low contrast')
plt.plot(right_high, label='Right high contrast')
plt.legend()
plt.xlabel('Time (ms)')
plt.ylabel('Firing rate (Hz)')
plt.title('Population average firing rate')
plt.show()
```

This code computes the average firing rate across all neurons and trials for each time point using np.mean(). Then it separates the spike data for different stimulus types using oolean indexing with vis_right. It computes the average firing rate for each combination of response and stimulus using np.mean() and oolean indexing with response and vis_right. Finally, it plots the results using plt.plot() and plt.legend(). The x-axis is time in milliseconds, and the y-axis is firing rate in Hz.

Figure:



5) The Peri-Stimulus Time Histogram (PSTH)

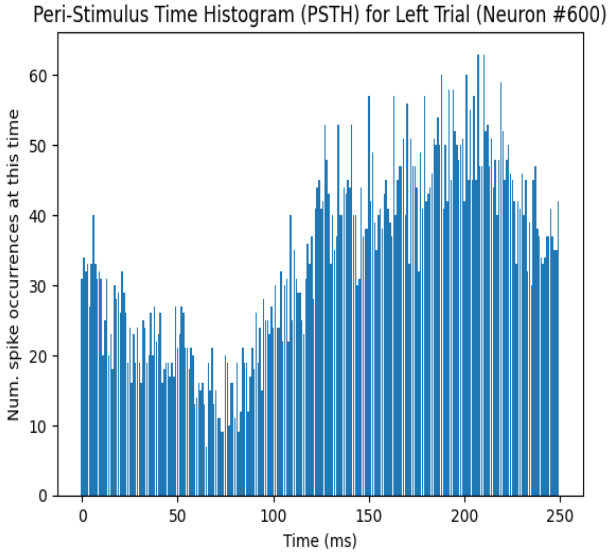
In this code `plot_histogram` is a function that takes in `spike_data` and title as arguments.

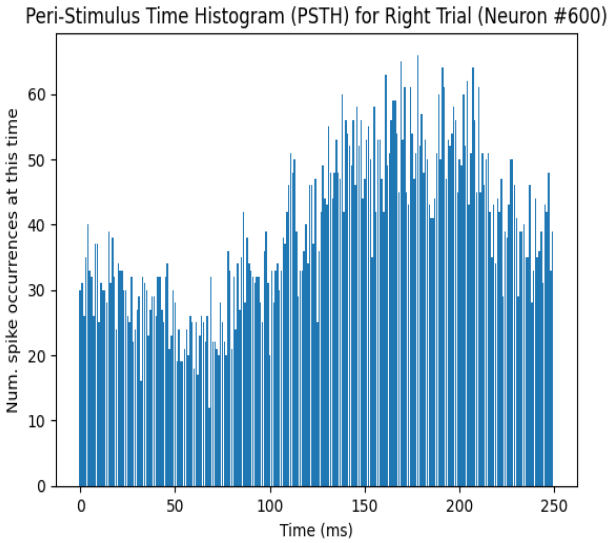
```
# MAIN CODE AND THE ANSWER FOR THIS PART OF EXERCISE!!!
# Separate the spike data for different trial types
NoGo_trial = dat['spks'][600, response == 0, :]
Left_trial = dat['spks'][600, response == 1, :]
Right_trial = dat['spks'][600, response == -1, :]
#print(NoGo_trial)
def plot_histogram(spike_data, title):
  fig, ax = plt.subplots()
  # Draw the PSTH
  ax.bar(range(spike_data.shape[1]),
         np.sum(spike_data, 0)
         )
  # Make pretty
  ax.set_title('Peri-Stimulus Time Histogram (PSTH) for '+title)
  ax.set_xlabel('Time (ms)')
  ax.set_ylabel('Num. spike occurrences at this time')
  plt.show()

plot_histogram(NoGo_trial, 'No-Go Trial (Neuron #600)')
plot_histogram(Left_trial, 'Left Trial (Neuron #600)')
plot_histogram(Right_trial, 'Right Trial (Neuron #600)')
```

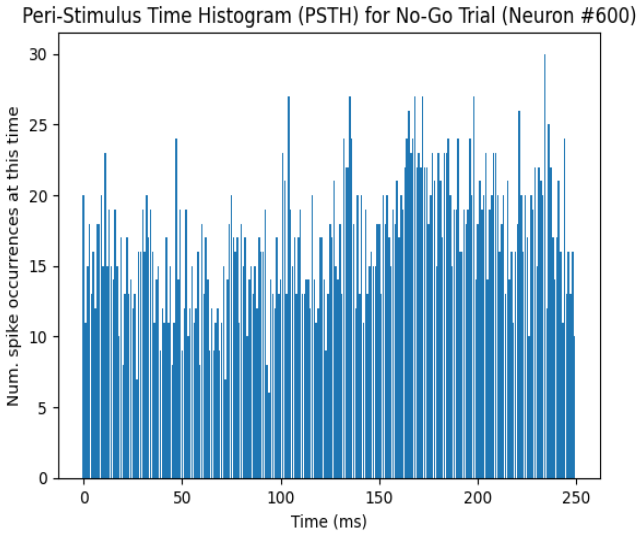
- `ax.bar()` creates a bar chart of the spike counts across time.
- `range(spike_data.shape[1])` creates an array of indices corresponding to the time bins.
- `np.sum(spike_data, 0)` sums the spike counts across trials. This calculates the Peri-Stimulus Time Histogram (PSTH) for the neuron and trial type of interest.

Those 3 final lines call `plot_histogram` three times with different input variables to plot the PSTHs for the different trial types.

Figures:





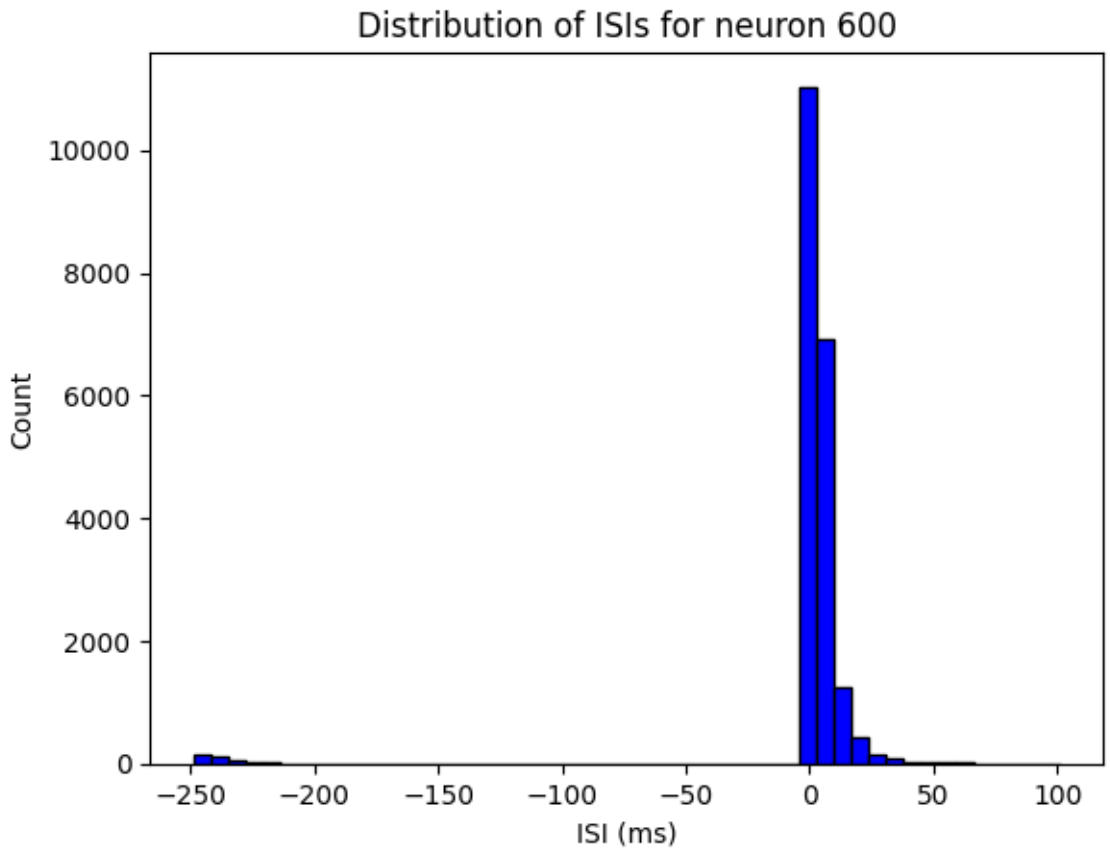


6) Inter-spike intervals and their distributions

```
# Step 1: Extract spike times for one neuron
spike_data = dat['spks'][600]  # select neuron 600
spike_times = np.argwhere(spike_data)[:, 1]  # extract spike times

# Step 2: Compute ISIs
isi = np.diff(spike_times)

# Step 3: Plot histogram of ISIs
plt.hist(isi, color='blue', edgecolor='black', bins=50)
plt.title('Distribution of ISIs for neuron 600')
plt.xlabel('ISI (ms)')
plt.ylabel('Count')
plt.show()
```

The code was pretty easy so I think there is no need for extra explanation except for the comments in the code!

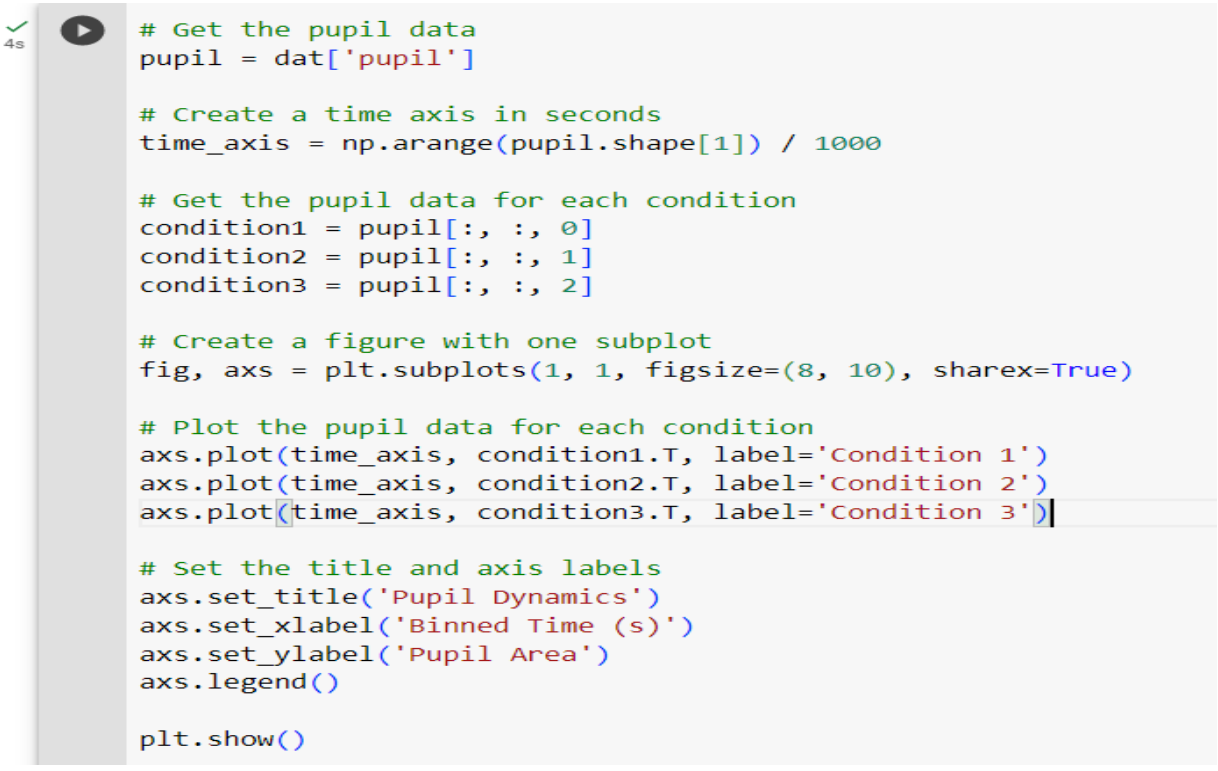
Here's the figure of the histogram plot that shows distribution of ISIs for neuron 600:



7) Behavioral data

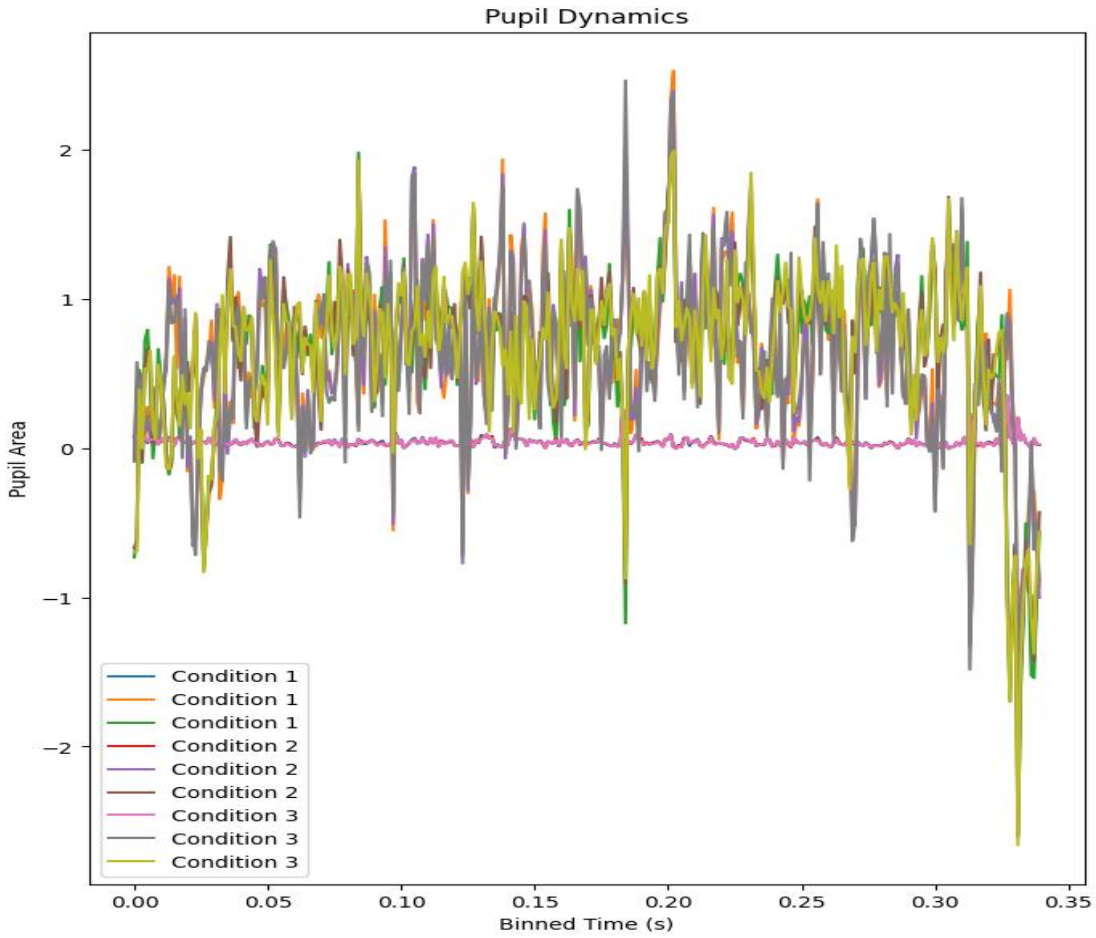
Pupil dilation represents the change in the size of the pupil, which is controlled by the autonomic nervous system. Specifically, the sympathetic nervous system causes the pupil to dilate (enlarge), while the parasympathetic nervous system causes it to constrict (shrink). Pupil dilation can be used as an indicator of cognitive and emotional processes, such as arousal, attention, and interest.

There may be a correlation between behavioral data (pupil dilation) and neuronal spikes, depending on the specific experimental paradigm and research question. For example, if the stimuli are designed to elicit a particular neural response (e.g., visual or auditory), then changes in pupil dilation may reflect changes in neural activity. However, this correlation is not always straightforward and may depend on various factors, such as the type of neurons recorded, the location of the recording electrodes, and the specific task demands. Further analysis and interpretation are needed to determine the relationship between behavioral data and neuronal spikes.



- condition1 = pupil[:, :, 0] : This line extracts the pupil data for the first condition from pupil and assigns it to the variable condition1. The first two dimensions of pupil correspond to the trial number and time bin, respectively, and the third dimension corresponds to the condition.(same goes for other conditions)
- -axs.plot(time_axis, condition1.T, label='Condition 1'): This line plots the pupil data for the first condition on the subplot using axs.plot. The time axis is plotted on the x-axis, and the pupil data is transposed using .T so that each trial is plotted as a separate line. The label for the line is set to 'Condition 1' using the label argument.

Figure:



Note! There's some extra code in the file which I found or write during solving the main questions; don't notice them!

PART 2: Theory Exercise

1.1)

$$f_X(x;\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} exp\left\{-\frac{x^2}{2\sigma^2}\right\}$$

$$E(X^2) = Var(X) + (EX)^2 = \sigma\,\hat{}\,2 + \mu\,\hat{}\,2$$

The likelihood function is

$$L(x;\sigma^2) = f_X(x;\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(x)^2}$$

The log-likelihood function is

$$\ln L(x;\sigma^2) = -\ln(2\pi)^{\frac{1}{2}} - \ln\sigma - \frac{x^2}{2\sigma^2}.$$

To find the MLE for σ, we differentiate $\ln L(x;\sigma^2)$ with respect to σ and set it equal to zero.

$$\begin{aligned}\frac{\partial}{\partial\sigma}\ln L &= -\frac{1}{\sigma} + \frac{x^2}{\sigma^3} \\ &= -\frac{1}{\sigma} + \frac{x^2}{\sigma^3} \overset{set}{=} 0.\end{aligned}$$

Therefore,

$$\hat{\sigma}X^2 = \hat{\sigma}^3 \rightarrow \hat{\sigma} = |X|.$$

Also, we can verify that the second derivative is negative to make sure that $\hat{\sigma} = |X|$ is actually the maximizing value:

$$\frac{\partial^2}{\partial\sigma^2}\ln L = \frac{1}{\sigma^2} - \frac{3x^2}{\sigma^4} < 0 \text{ when } \hat{\sigma} = |x|.$$

1.2)

$$X \sim N(\mu, \tau^2) \quad \text{and} \quad Y \mid X = x \sim N(x, \sigma^2),$$

it can be shown that the posterior density of X given $Y = y$ is given by

$$X \mid Y = y \sim N\left(\frac{y/\sigma^2 + \mu/\tau^2}{1/\sigma^2 + 1/\tau^2}, \frac{1}{1/\sigma^2 + 1/\tau^2}\right).$$

Since $Y \mid X = x \sim N(x, \sigma^2)$, we conclude

$$\overline{Y} \mid X = x \sim N\left(x, \frac{\sigma^2}{n}\right).$$

Therefore, we can use the posterior density given in the problem statement (we need to replace σ^2 by $\frac{\sigma^2}{n}$). Thus, the posterior density of X given $\overline{Y}$ is

$$X \mid \overline{Y} \sim N\left(\frac{n\overline{Y}/\sigma^2 + \mu/\tau^2}{n/\sigma^2 + 1/\tau^2}, \frac{1}{n/\sigma^2 + 1/\tau^2}\right).$$

To find the MAP estimate of X given $\overline{Y}$, we need to find the value that maximizes the posterior density. Since the posterior density is normal, the maximum value is obtained at the mean which is

$$\hat{X}_{MAP} = \frac{n\overline{Y}/\sigma^2 + \mu/\tau^2}{n/\sigma^2 + 1/\tau^2}.$$

(1.3

هرچه تعداد نمونه‌های جامعه بیشتر شود، تخمین ML و تخمین MAP به تخمینی شبیه به هم نزدیک‌تر خواهند شد، و ارتباط بین آن‌ها نیز نزدیک‌تر خواهد بود .به طور کلی، هنگامی که تعداد نمونه‌ها بسیار بیشتر از اندازه پارامترهاست، تخمین ML و تخمین MAP به یکدیگر نزدیک می‌شوند و برای تخمین‌های منحنی یادگیری نیز جمعیت بزرگ کافی است. با این حال، در صورتی که تعداد نمونه‌ها بسیار کم باشد، تخمین ML و تخمین MAP با هم فاصله دارند و می‌توانند به نتایج متفاوتی منجر شوند .به عنوان مثال، در صورتی که تعداد نمونه‌ها کمتر از ۳ تا ۵ برابر اندازه پارامتر باشد، تخمین MAP با میانگین پیشین پارامتر (prior) گرافیکی اولویتی دارد و پارامتر بیشتری را تخمین می‌زند. در برخی موارد، این ممکن است دلایلی مثل شکل خروجی و غیره را مد نظر داشته باشد. با افزایش تعداد نمونه ها، تخمین MAP با تخمین ML همگرا خواهند شد و این تاثیر پریور نیز در حداقل رفتن خطاهای تخمین کاهش خواهد یافت.

(1.4

قانون توان در یادگیری می‌گوید که افزایش توان و فعالیت نورون‌ها در سطوح بالاتر از یادگیری، به علت افزایش تعداد اتصالات بین نورون‌ها، تقویت شدن اتصالات پیش‌رو و رشد دستگاه‌های سیتوپلاسمی است. به عبارت دیگر، هرچه بیشتر درس خوانده و مهارت‌های جدید یاد گرفته شود، تعداد اتصالات بین نورون‌ها افزایش می‌یابد و فعالیت نورون‌ها تقویت می‌شود که منجر به افزایش توان نورونی می‌شود .

با توجه به قانون توان، انجام تمرین و یادگیری بهتر منجر به افزایش توان و فعالیت نورون‌ها خواهد شد، زیرا با انجام تمرین های بیشتر، اتصالات بین نورون ها تقویت می‌شوند و فعالیت نورون ها افزایش می‌یابد .

در مورد قسمت دوم سوال، اگر یک مهارت یکبار کسب شود و فراموش شود، به نظر می‌رسد که اتصالات بین نورون‌ها قطع نمی‌شوند و توان نورونی نیز برخلاف فرآیند فراموشی که با کاهش تعداد اتصالات و کاهش توان نورونی همراه است، کاهش نمی‌یابد. بنابراین، با دوباره کسب مهارت، نورون‌ها از قانون قبلی تبعیت خواهند کرد و فعالیت نورون‌ها و توان نورونی مجدداً افزایش پیدا می‌کند .

برای آزمایش خواندن متن معکوس، می‌توان از طرحی استفاده کرد که به طور خلاصه به شرح زیر است:

- گروه آزمایشی را به دو دسته تقسیم کرده و به یکی از دسته‌ها یاد داده می‌شود که متون معکوس شده را بخوانند .
- پس از یادگیری، با استفاده از تکنیک های تصویرسازی دماغ و فعالیت بر روی محلول های شیمیایی، تحلیل تصویرسازی توزیع فعالیت نورون‌های گروه آزمایشی انجام می‌شود .
- سپس، توان نورون ها و فعالیتشان در هنگام خواندن متن معکوس و خواندن متن معمولی مقایسه می‌شود تا مشخص شود آیا خواندن متن معکوس، توان نورون‌ها را افزایش می‌دهد یا نه و آیا توان نورون‌ها بعد از یادگیری مجدد مقداری کاهش پیدا می‌کند یا نه.