Sharif university of technology
Faculty of electrical engineering

Neuroscience ,learning ,memory and cognition

Dr. karbalaie

Soraya Charkas
99101387

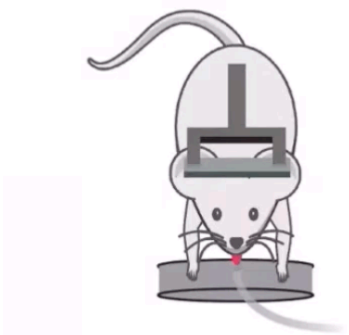
HomeWork 3

June 6, 2023

Computer Assignment

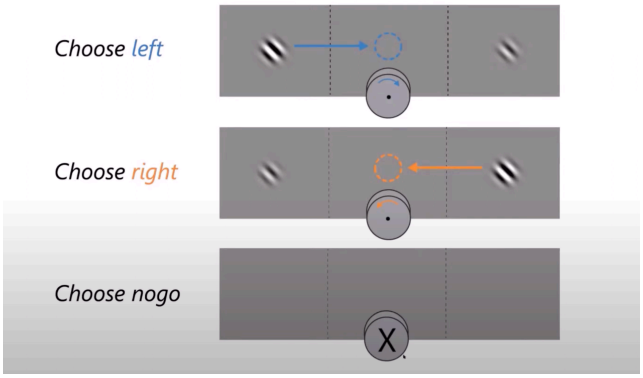
Exersice outline

The experiment involves a visual contrast detection and discrimination task for mice, facilitated by the YouTube video. The mice are situated in an environment surrounded by three computer screens to provide visual stimuli. To perform the task, the mice are required to manipulate a rubber wheel using their four limbs, either in a clockwise or counterclockwise direction. As a reward, water is provided to incentivize and train the mice. Two specific brain regions, the secondary motor cortex anterior cingulate and the primary visual parts of the hippocampus, are targeted for recording neural activity.

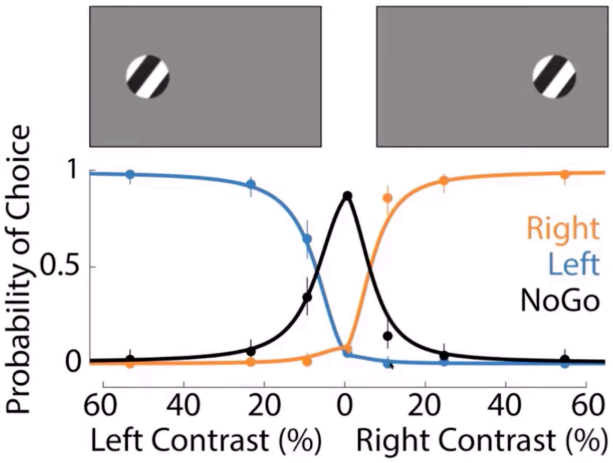


Prob setup

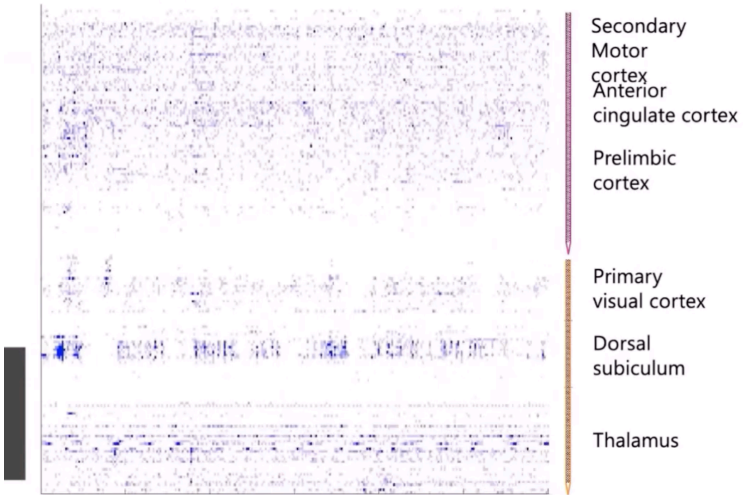
during the experiment. A total of 10 mice were tested, resulting in 39 recording sessions and the submission of data from approximately 30,000 neurons. The experimental setup includes different conditions, such as the presence of light on the right side, which requires clockwise turning, or the opposite condition. Additionally, there are instances where no wheel rotation is needed, referred to as the "nogo" condition.



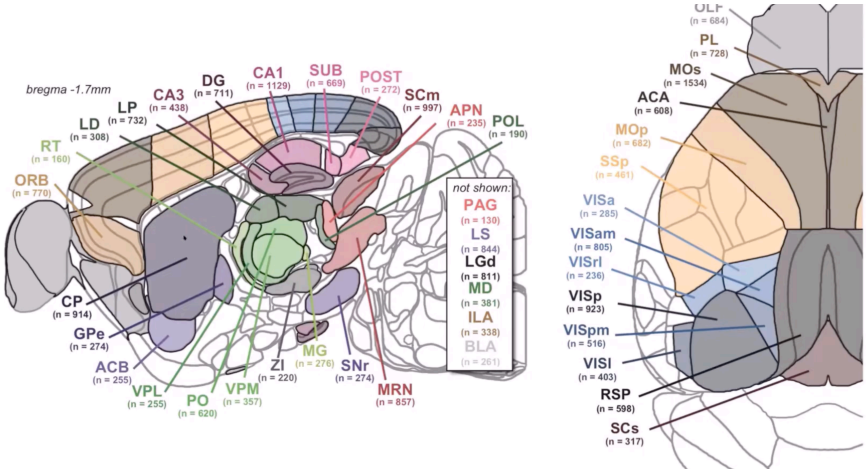
The probability of choises the mouse has choosen are as below :



Also, as we no the raster plot is a figure which can show us the spikes during different sessions and are valuable because we can see a specific pattern of spikes for a certain stimuli :



Throughout the experiment, neural recordings were obtained from 42 distinct brain regions which are as below :



Load the given dataset

Using the Jupyter notebook, I loaded the data as below:

```
with open('Visual_Cortex_Stinmetz_Dataset.p', 'rb') as fp:
    dat = pickle.load(fp)
    print(dat['mouse_name'])
```

Also the mice name was Lederberg.

Raster plot

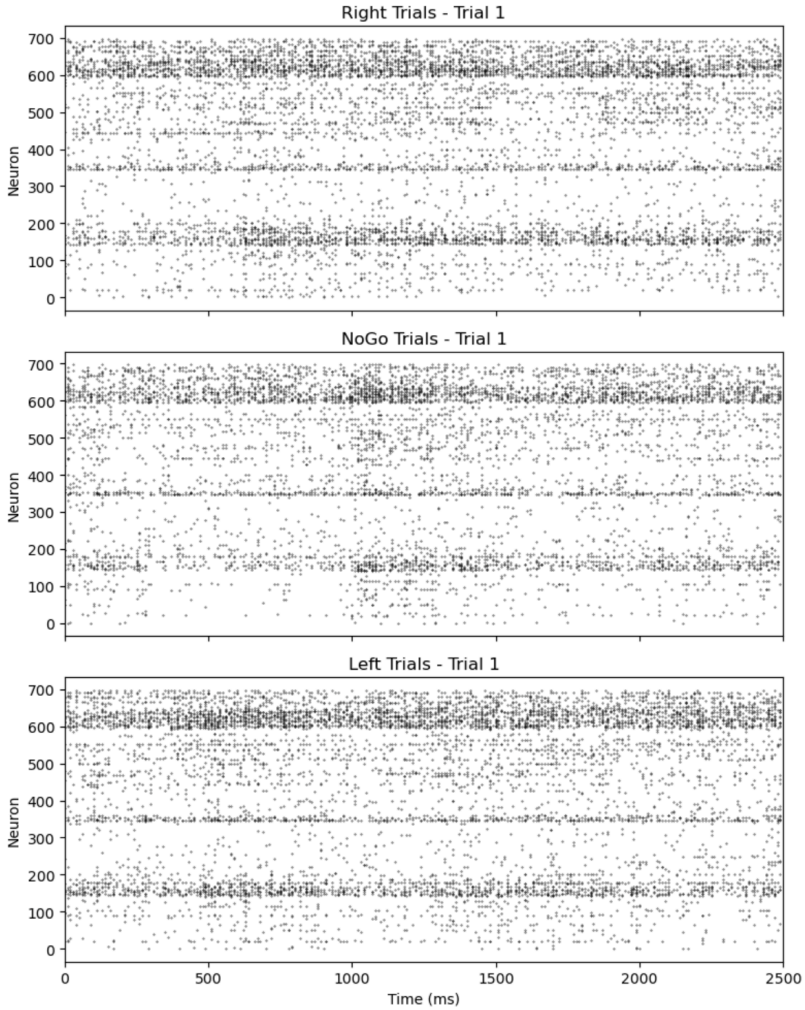
At first , we were supposed to count the number of left,right and Nogo trials. I wrote a simple code to count the number of each parameter :

```
In [25]: dt = dat['bin_size']  # binning at 10 ms
         NT = dat['spks'].shape[-1] # number of trials
         def count_trials(array):
             right_count = 0
             nogo_count = 0
             left_count = 0

             for value in array:
                 if value == 1:
                     right_count += 1
                 elif value == 0:
                     nogo_count += 1
                 elif value == -1:
                     left_count += 1

             return right_count, nogo_count, left_count
         count_trials(response)

Out[25]: (135, 64, 141)
```

After counting the number of each trial , we are ready to plot the raster plot. I have put some comments so that the code be readable, but this is a short explain of what it does.as we know and as I explaind above, in raster plot we are plotting the spikes with respect to their place and for different sessions, we are hoping to see a pattern and to see that a specific spike at a specific time.for each trial we only plot a dot using scatter plot every time there is a spike .the code is as below :

```
# NoGo_trial = dat['spks'][:, response == 0,:]
Left_trial = dat['spks'][:, response == 1,:]
Right_trial = dat['spks'][:, response == -1,:]
# Select a specific trial for visualization
trial_index = 0

# Separate spike data for each trial type
right_spikes = Right_trial
nogo_spikes = NoGo_trial
left_spikes = Left_trial

# Get the number of neurons
num_neurons = right_spikes.shape[0]

# Set the time range
time_bins = np.arange(0, 2500, 10)

# Plot the raster for each trial type
fig, axs = plt.subplots(3, 1 , figsize=(10,10), sharex=True)

# Plot for Right trials
for neuron in range(num_neurons):
    spike_times = np.where(right_spikes[neuron, trial_index, :] > 0)[0]
    axs[0].scatter(time_bins[spike_times], [neuron] * len(spike_times), color='black', s=0.1)
axs[0].set_ylabel('Neuron')
axs[0].set_title(f'Right Trials - Trial {trial_index+1}')

# Plot for NoGo trials
for neuron in range(num_neurons):
    spike_times = np.where(nogo_spikes[neuron, trial_index, :] > 0)[0]
    axs[1].scatter(time_bins[spike_times], [neuron] * len(spike_times), color='black', s=0.1)
axs[1].set_ylabel('Neuron')
axs[1].set_title(f'NoGo Trials - Trial {trial_index+1}')

# Plot for Left trials
for neuron in range(num_neurons):
    spike_times = np.where(left_spikes[neuron, trial_index, :] > 0)[0]
    axs[2].scatter(time_bins[spike_times], [neuron] * len(spike_times), color='black', s=0.1)
axs[2].set_xlabel('Time (ms)')
axs[2].set_ylabel('Neuron')
axs[2].set_title(f'Left Trials - Trial {trial_index+1}')
plt.xlim(0, 2500)  # Set the x-axis limits to 0 to 2.5 ms
plt.tight_layout()
plt.show()
```

And the output of raster plots for left, right and Nogo trials are as below :



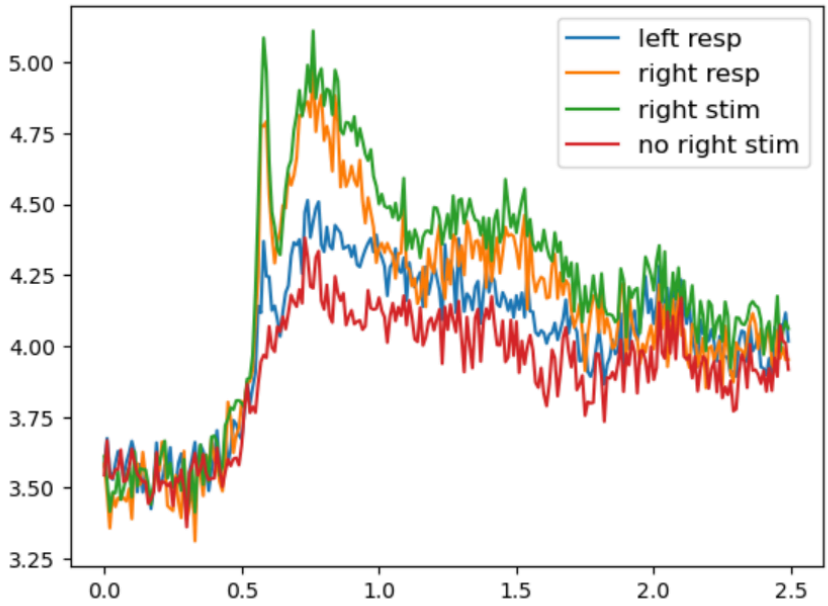
As we expected, we can see a pattern , in some areas the figure is more highkighted wich means more spike or in better words spikes in a specific place.

Basic population average

The. Code written for this part, generally visualizes the firing rates of neurons in response to different experimental conditions, including left and right responses, presence or absence of stimuli on the right, and the overall average firing rate.

Each line that includes the code plot.plt is plotting the average firing rate of a neuron for a condition as said in the code. The code and output are as below:

```
response = dat['response']  # right - nogo - left (-1, 0, 1)
vis_right = dat['contrast_right']  # 0 - low - high
vis_left = dat['contrast_left']
plt.plot(dt * np.arange(NT), 1/dt * dat['spks'][:, response >= 0].mean(axis=(0, 1)))
plt.plot(dt * np.arange(NT), 1/dt * dat['spks'][:, response < 0].mean(axis=(0, 1)))
plt.plot(dt * np.arange(NT), 1/dt * dat['spks'][:, vis_right > 0].mean(axis=(0, 1)))
plt.plot(dt * np.arange(NT), 1/dt * dat['spks'][:, vis_right == 0].mean(axis=(0, 1)))

plt.legend(['left resp', 'right resp', 'right stim', 'no right stim'], fontsize=12)
ax.set(xlabel='time (sec)', ylabel='firing rate (Hz)')
plt.show()
```



The Peri-Stimulus Time Histogram (PSTH)

The PSTH plots provide a visualization of the average spike activity of neurons across different trial types (Right, NoGo, and Left) as a function of time. By aggregating the spike counts within short time bins, we can observe the temporal patterns of neural activity in response to different stimuli.

From the PSTH plots, we can make several observations and discuss the results:

-Right Trials: The PSTH plot for the Right trials (blue) shows the average spike count as a function of time. It provides an indication of when neurons are most likely to spike during the presentation of the Right stimulus. If there is a significant increase in spike activity at specific time intervals, it suggests that neurons are selectively responding to the Right stimulus.

-NoGo Trials: The PSTH plot for the NoGo trials (green) represents the average spike count as a function of time when there is no response made by the mouse. Since the mouse does not make a response in NoGo trials, the absence of a motor response can still elicit neural activity. The PSTH can reveal if there are any time intervals where neurons exhibit increased or decreased activity during NoGo trials compared to baseline activity.

-Left Trials: The PSTH plot for the Left trials (red) displays the average spike count as a function of time for the Left stimulus. Similar to the Right trials, this plot helps identify when neurons are most likely to respond to the Left stimulus.

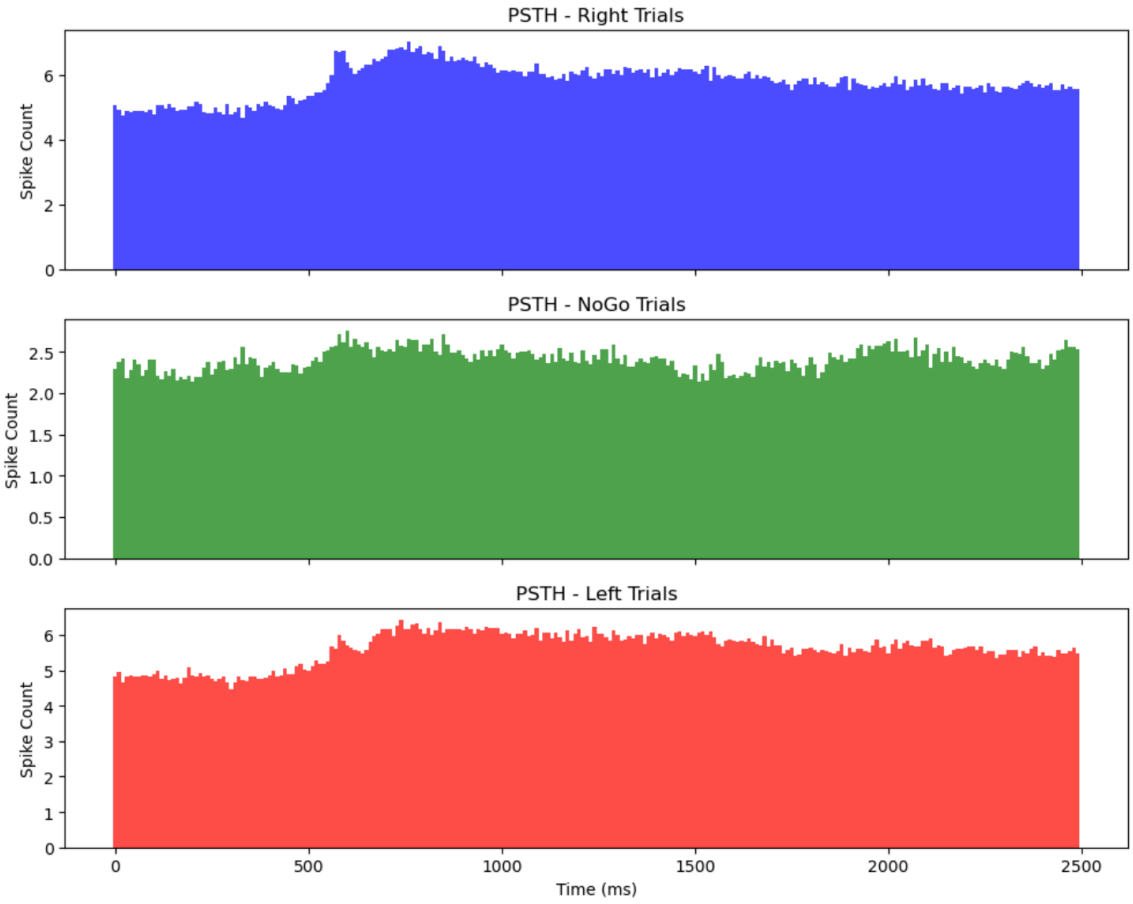
The important part of the code is where we calculate the PSTH for each trial:

```
# Calculate the PSTH for each trial type
right_psth = np.sum(right_spikes, axis=(0, 1)) / num_neurons
nogo_psth = np.sum(nogo_spikes, axis=(0, 1)) / num_neurons
left_psth = np.sum(left_spikes, axis=(0, 1)) / num_neurons
```

The complete code and the output plots are as below :

```
right_spikes = Right_trial
nogo_spikes = NoGo_trial
left_spikes = Left_trial

num_neurons = right_spikes.shape[0]
num_time_bins = right_spikes.shape[2]

# Calculate the PSTH for each trial type
right_psth = np.sum(right_spikes, axis=(0, 1)) / num_neurons
nogo_psth = np.sum(nogo_spikes, axis=(0, 1)) / num_neurons
left_psth = np.sum(left_spikes, axis=(0, 1)) / num_neurons


time_bins = np.arange(0, num_time_bins * time_bin_size, time_bin_size)

# Plot the PSTH
fig, axs = plt.subplots(3, 1, figsize=(10, 8), sharex=True)

axs[0].bar(time_bins, right_psth, width=time_bin_size, color='blue', alpha=0.7)
axs[0].set_ylabel('Spike Count')
axs[0].set_title('PSTH - Right Trials')

axs[1].bar(time_bins, nogo_psth, width=time_bin_size, color='green', alpha=0.7)
axs[1].set_ylabel('Spike Count')
axs[1].set_title('PSTH - NoGo Trials')

axs[2].bar(time_bins, left_psth, width=time_bin_size, color='red', alpha=0.7)
axs[2].set_xlabel('Time (ms)')
axs[2].set_ylabel('Spike Count')
axs[2].set_title('PSTH - Left Trials')

plt.tight_layout()
plt.show()
```

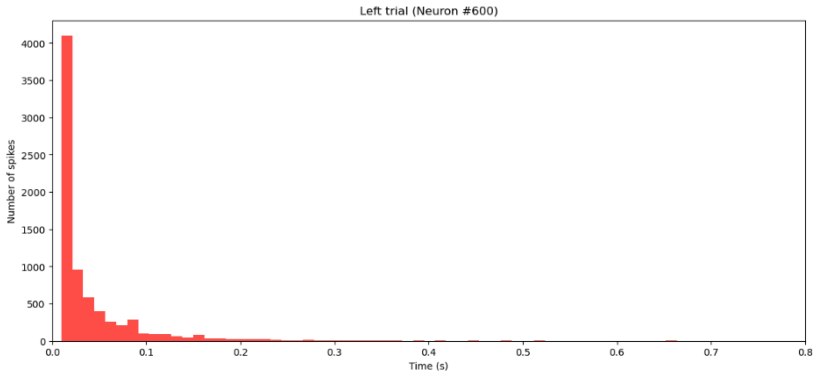
After running this code, we get the outputs as below :

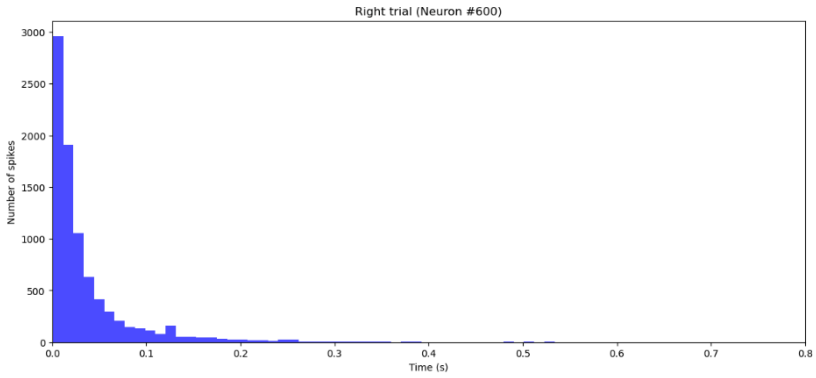


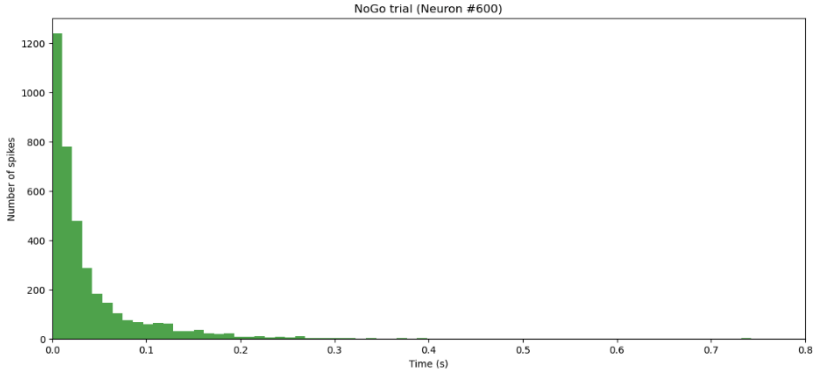
Note that my timespace is 0-2500 thats why my plots are a little crowded.

Inter-spike intervals and their distributions

Using the steps said in the assignment and implementing it with python, I got theser outputs(the colours for each trial is the same as the previous part)







The code is an efficient and simple implementation of the hints given in the questions :

```
neuron = 600
trial_types = [Left_trial, Right_trial, NoGo_trial]
colors = ['red','blue', 'green']
titles = ['Left trial', 'Right trial', 'NoGo trial']

for i, trial_type in enumerate(trial_types):
    neuron_trial = trial_type[neuron, :, :]
    neuron_trial[neuron_trial > 0] = 1
    neuron_trial = neuron_trial.astype(float)
    for j in range(trial_type.shape[1]):
        for k in range(trial_type.shape[2]):
            neuron_trial[j][k] = neuron_trial[j][k] * t[k]
    neuron_trial = neuron_trial[neuron_trial != 0]
    inter_spike_intervals = np.diff(neuron_trial)

    plt.figure(figsize=(14, 6))
    plt.hist(inter_spike_intervals, bins=300, color=colors[i],alpha = 0.7)
    plt.title(f'{titles[i]} (Neuron #{neuron})')
    plt.xlabel('Time (s)')
    plt.ylabel('Number of spikes')
    plt.xlim(0, 0.8)
    plt.show()
```

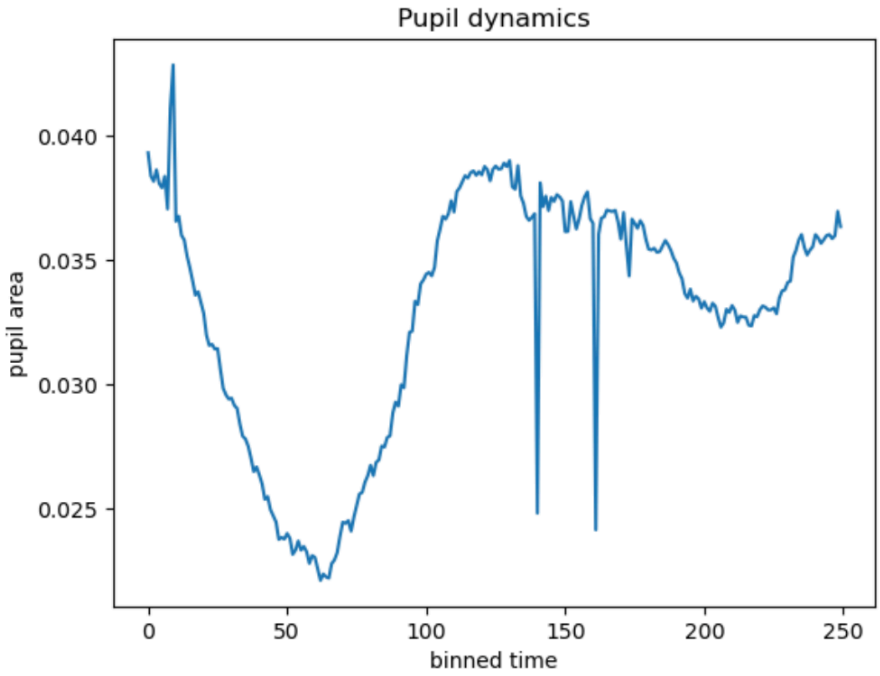
Behavioral data

Pupil dilation can reflect changes in cognitive or emotional states. Generally, increased pupil dilation is associated with heightened arousal, attention, or cognitive effort. It can be influenced by factors such as visual stimulation, task demands, emotional valence, and cognitive load. On the other hand, decreased pupil dilation can indicate reduced arousal or attention.Regarding the correlation between behavioral data (pupil) and neuronal spikes, it is possible to explore the relationship between these two variables. By aligning the neuronal activity with the behavioral data, such as the pupil area, it is possible to examine whether there are temporal associations between changes in neuronal firing rates and pupil dilation.one can compute the average neuronal firing rate during specific time periods, such as when the pupil is dilating or constricting. Statistical analyses, such as correlation or regression analyses, can then be performed to investigate the potential relationship between pupil dilation and neural activity.

It is important to note that while correlations between behavioral data and neuronal spikes can provide insights into potential relationships, they do not necessarily imply causation.

```
ax = plt.subplot(1,1, 1)
plt.plot(dat['pupil'][0, :].mean(0));
ax.set(ylabel='pupil area', xlabel='binned time', title='Pupil dynamics')
```

```
[Text(0, 0.5, 'pupil area'),
 Text(0.5, 0, 'binned time'),
 Text(0.5, 1.0, 'Pupil dynamics')]
```



The written code and the output are :

Bonus

I could not download the files so this part in incomplete.

Theory assignment

ML and MAP

Question one

we derive the likehood term :

$$P(x_1, \ldots, x_n \mid \mu) = \Pi_{i=1}^{N} P(x_i \mid \mu) = \Pi_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

We knoiw that log is a monotonically increase function so we maxsimise log of likehood :

$$log(P(x_1, \ldots x_N \mid \mu) = \sum_{i=1}^{N} log(\frac{1}{\sqrt{\pi\sigma^2}}) - \frac{(x_i-\mu)^2}{2\sigma^2}$$

We want to find μ so we take derivative of both sides with respect to μ and set it to zero :

$$\frac{dlog(P(x_1, \ldots x_N \mid \mu)}{d\mu} = \sum_{i=1}^{N} \frac{(x_i-\mu)^2}{2\sigma^2} => \sum_{i=1}^{N} \frac{(x_i-\mu)^2}{2\sigma^2} = 0 => \sum_{i=1}^{N} (x_i-\mu)^2 = 0 => \sum_{i=1}^{N} \mu = \sum_{i=1}^{N} x_i => N\mu = \sum_{i=1}^{N} x_i => \hat{\mu} = \frac{\sum_{i=1}^{N} x_i}{N}$$

Question two

We use the baye's rule :

$$P(\mu \mid x_1, \ldots, x_n) = \frac{P(x_1, \ldots, x_n \mid \mu) P(\mu)}{P(x_1, \ldots, x_n)}$$

As we obtained earlier , we know that

$$P(x_1, \ldots, x_n \mid \mu) = \Pi_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

Which means :

$$P(\mu) = \frac{1}{\sqrt{2\pi\beta^2}} e^{-\frac{(\mu-\nu)^2}{2\beta^2}}$$

So we are trying to find the value of μ which maximizes :

$$P(\mu \mid x_1, \ldots, x_n) = \frac{(\Pi_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}) \frac{1}{\sqrt{2\pi\beta^2}} e^{-\frac{(\mu-\nu)^2}{2\beta^2}}}{P(x_1, \ldots, x_n)}$$

Just like the first part , we can use log:

$$log(P(\mu \mid x_1, \ldots, x_n)) = (\sum_{i=1}^{N} - log(\sqrt{\pi\sigma^2}) - \frac{(x_i-\mu)^2}{2\sigma^2}) - log(\sqrt{2\pi\beta^2}) - \frac{(\mu-\nu)^2}{2\beta^2}$$

Once again we take derivative of both sides with respect to μ and set it to zero :

$$\frac{dlog(P(\mu \mid x_1, \ldots, x_n))}{d\mu} = (\sum_{i=1}^{N} \frac{(x_i-\mu)^2}{2\sigma^2}) - \frac{(\mu-\nu)^2}{2\beta^2}) = > 0 = (\sum_{i=1}^{N} \frac{(x_i-\mu)^2}{2\sigma^2}) - \frac{(\mu-\nu)^2}{2\beta^2})$$

$$\sum_{i=1}^{N} \frac{(x_i-\mu)^2}{2\sigma^2} = \frac{(\mu-\nu)^2}{2\beta^2} = > -\sum_{i=1}^{N} \frac{(x_i)}{2\sigma^2} - \frac{N\mu}{\sigma^2} = \frac{(\mu-\nu)}{\beta^2}$$

$$\hat{\mu} = \frac{\sigma^2\nu + \beta^2 \sum_{i=1}^{N} x_i}{\sigma^2 + N\beta^2}$$

We can see that this estimation is more accurate , but in next section we will seen that this statement is only true when N is small.

Question three

We saw that :

$\hat{\mu}_{ML} = \frac{\sum_{i=1}^{N} x_i}{N}$ and $\hat{\mu}_{MAP} = \frac{\sigma^2\nu + \beta^2 \sum_{i=1}^{N} x_i}{\sigma^2 + N\beta^2}$ it is obvious as we

increas N, the terms $\frac{\sigma^2\nu}{\sigma^2 + N\beta^2}$ are getting equal to zero , so as we increase

N the ML and MAP estimations are getting identical .

Stimulus-response power laws

The power law of skill acquisition, or the power law of practice, basically says that the more skilled you are at something, the slower you improve with more practice. This happens in sports, music, and thinking tasks.

When you practice and learn, it can make your brain stronger and more active. Deliberate practice, which means focused and structured efforts to get better, can change your neurons and make it work better. With practice, the connections in your neurons related to the skill get stronger, and your brain works more smoothly and efficiently.

If you learn something and then forget it, relearning it might not follow the same pattern as when you first learned it. Forgetting means the connections in your neurons for that skill have weakened or faded because you haven't used them. Relearning the skill means reactivating and strengthening those connections, which might take less time and effort than when you first learned it.

The experiment is as below:

We had people join in and split them into two groups: the experimental group and the control group. We wanted to see what happens when you practice reading text that's flipped the other way around.

First, we checked the reading skills of both groups before the experiment to see where they were starting from. Then we went into the training phase.

In the training phase, the experimental group learned how to read reversed text. They did different exercises and tasks that helped them read text that was flipped horizontally. The control group didn't do any training related to reading reversed text. They did other things that weren't connected to the experiment, just to make sure we didn't get confused by other factors.

After a while of training, we tested both groups again to see if their reading skills had improved when it came to reversed text.

Then we had a retention phase where both groups didn't practice or see any more reversed text for a specific time.

After that, we tested both groups one more time to see if they still remembered how to read reversed text.

We looked at the results to compare how well the experimental and control groups did in reading reversed text. We expected the experimental group to improve more than the control group. We also wanted to see if the experimental group could still remember the skill after not practicing for a while. This would tell us if the skill could be regained even if you didn't practice it regularly.

END