Soroush Heidary 96222031

Exercise 2

Part 1 :

A population of 800 excitatory and 200 inhibitory neurons

I have created a class named population which creates the neurons and their synapses of the whole population, other methods of this class take care of simulating our neurons step by step so we can handle the spiking of each neuron and transforming it to an Input Current for the post synaptic neurons connected to it.

(There are 3 files in the rar I sent, Models are the exact code of phase 1 of the project, Neuron is the file containing the class we're gonna use (a modified version of Models so to be more convenient with the population properties and Population file which is shown below …)

There are also 2 other methods which calculate population activity, and raster plot of the population

Get_activity() :

This function procedes through this formula :

$$A(t) = \lim_{\Delta t \to 0} \frac{1}{\Delta t} \frac{n_{\text{act}}(t; t+\Delta t)}{N} = \frac{1}{N} \sum_{j=1}^{N} \sum_{f} \delta\left(t - t_j^{(f)}\right) ,$$

The above formula is basically calculating the number of spikes in a little range of 't', divided by the size of the population and further more by the range of 't' itself,

for the implementation I considered this t to be equal to 5 time steps (around 1.5 millisecons)

this is the code which does the explained procedure :

```
for j in range(len(self.neurons)) :
    temp = np.array(self.neurons[j].spike_history)
    temp = temp[temp <= t]
    temp = temp[temp >= t-delta_t]
    action_sum += len(temp)

activity[t] = action_sum / delta_t / self.size
```

Raster_plot() :

This function doesn't need much explaination as it does only calculates all the spikes that have been occurred in time step 't' and plots them

The inhibitory neurons are shown with blue dots while excitatory neurons are show as orange dots

Synaps creation in the class creates synapses by a probability chance of connection which is an initial input of __ini__ function, this chance would be equal to 1 when treating with fully connected populations, here's the code :

```
def Synaps_creation(self) :
    chance = int(self.size * self.connection_ratio)

    for i in range(len(self.neurons)) :
        con_list = random.sample(range(0, self.size - 1), chance)
        if i in con_list :
            con_list.remove(i)
        self.neurons[i].post_synapses = con_list
```

The process of simulating the neurons happens on two distinct functions : 'Next_tick' and 'Adjust_current' :

Suggesting by the names the first one runs each of the neurons of the population by one tick, and after that the second function checks if any of the neurons spiked, if

so, it transfers the Input Current generated by the spike to the post synaptic neurons connected to it.

How the spike becomes an input is simply a fixed float number (which actually represents the strength too)

This 2 functions get called by the amount of time steps (which is equal to 3000 here)

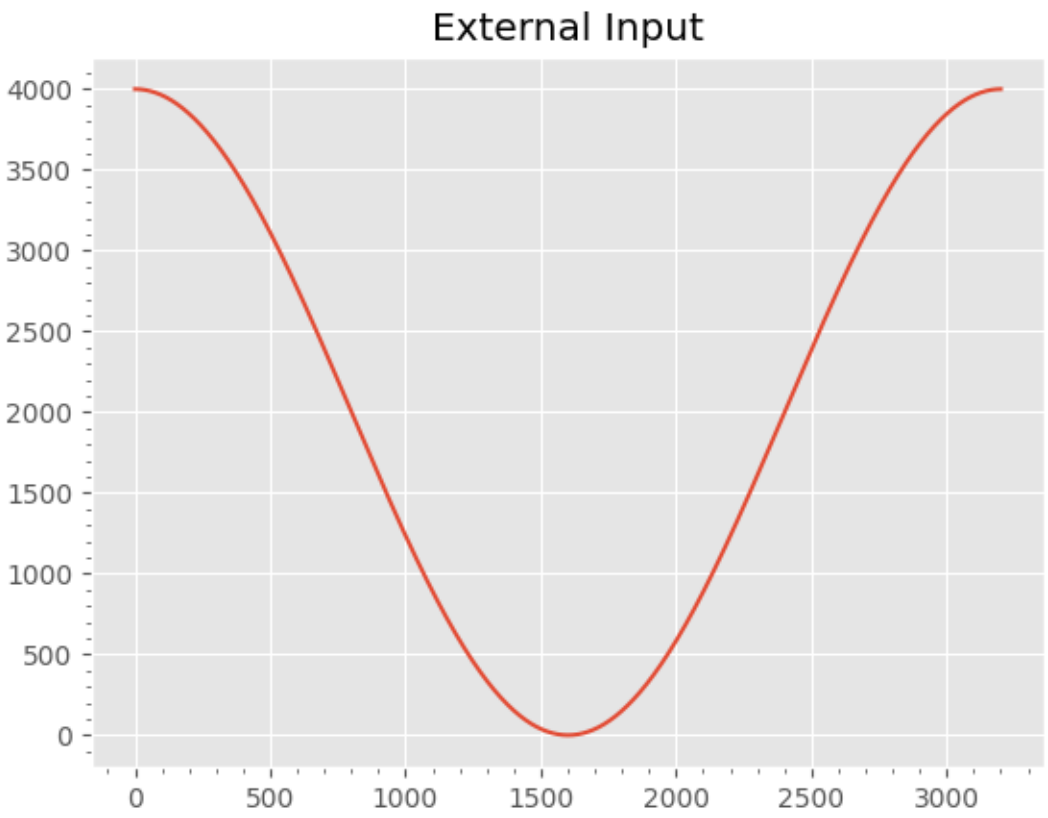
Here are the codes :

```
def Next_tick(self, t) :
    for neuron in self.neurons :
        neuron.Simulate_tick(t)


def Adjust_current(self, t) :
    for neuron in self.neurons :
        if neuron.just_spiked :
            for j in neuron.post_synapses :
                temp = (0.4  + random.randint(0, 10)/100) * neuron.type
                self.neurons[j].I[t+1] += temp
                self.I_in[t+1] += temp
```

A random noise is added to each time we're transferring input currents to avoid having identical results, also the weight or strength of a synaps is set to a fixed number here and its also important to note that inhibitory neurons transfer a negative amount of current shown here by the neuron type (inhibitory neurons have type = -1)

Results :

We’re going to use this External input for our population :



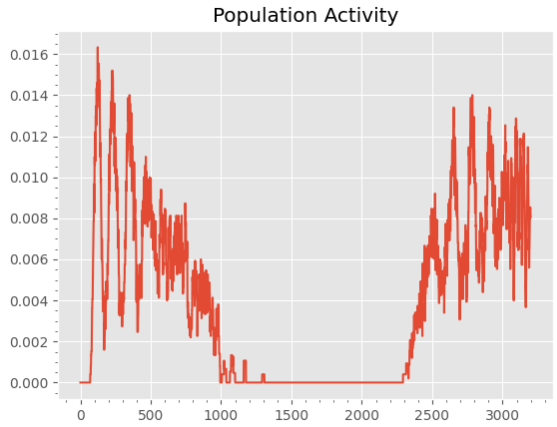
The 4 plots shown here are :

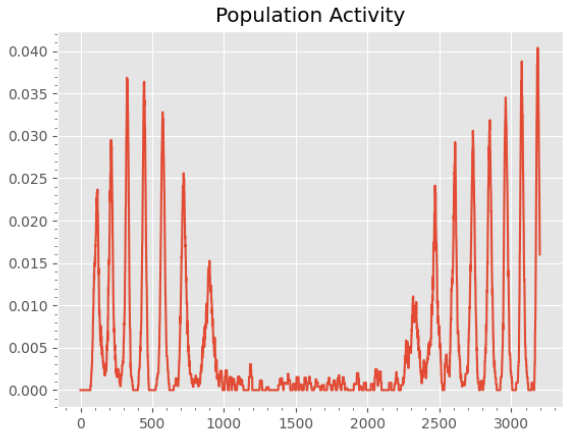
Population activity, Raster plot, Internal Input of population, Voltage of a single neuron (neuron number 42)

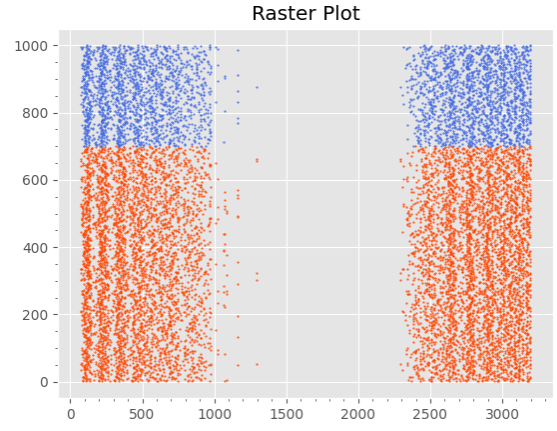
(Respectively from top to down)

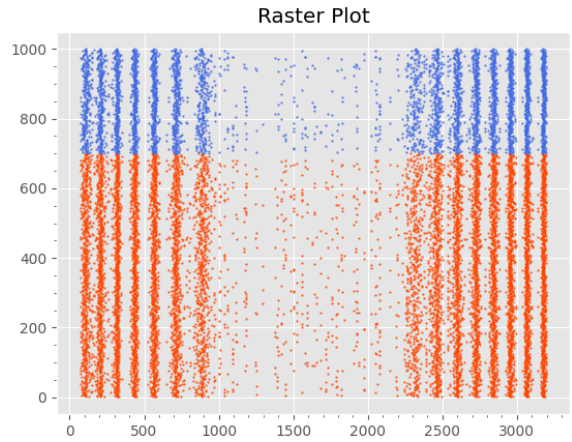
Left column (1st set of code output) is when have weaker synapses

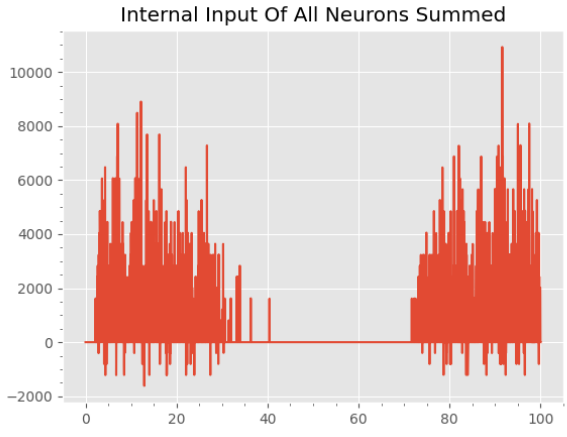
And the Right column (2nd set of code output) is when we have stronger synaptic connections

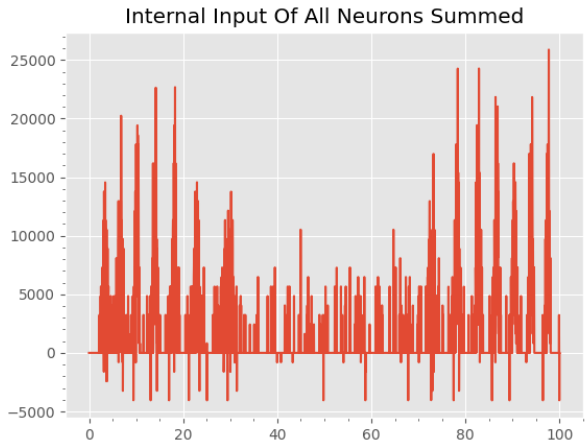

Population Activity
0.016
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
0
500
1000
1500
2000
2500
3000

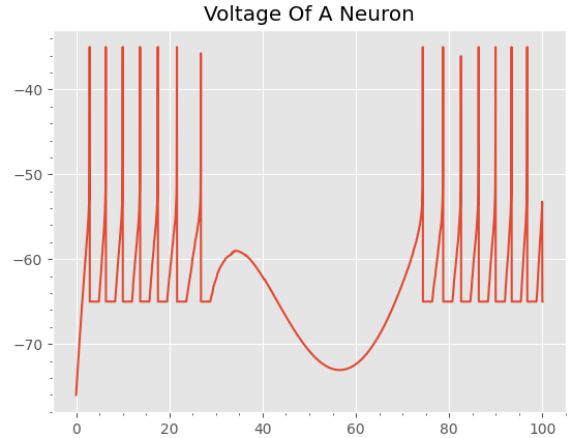

Population Activity
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
500
1000
1500
2000
2500
3000

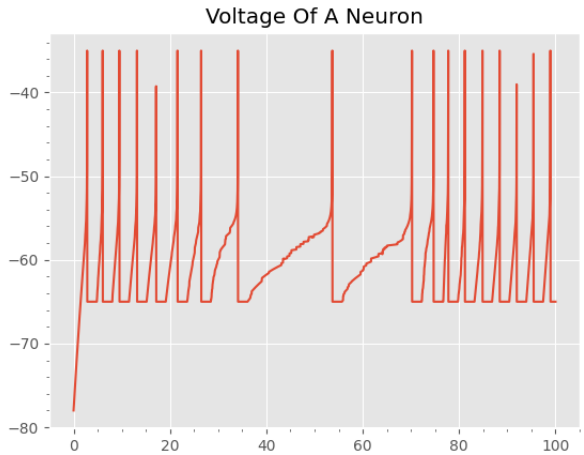

Raster Plot
1000
800
600
400
200
0
0
500
1000
1500
2000
2500
3000


Raster Plot
1000
800
600
400
200
0
0
500
1000
1500
2000
2500
3000


Internal Input Of All Neurons Summed
10000
8000
6000
4000
2000
0
−2000
0
20
40
60
80
100

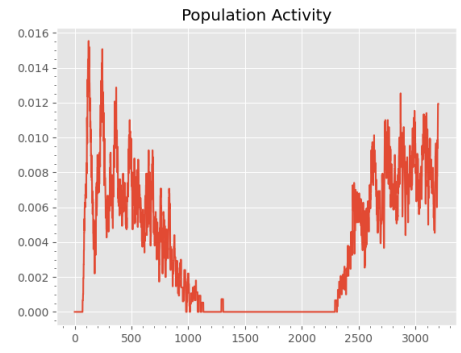

Internal Input Of All Neurons Summed
25000
20000
15000
10000
5000
0
−5000
0
20
40
60
80
100

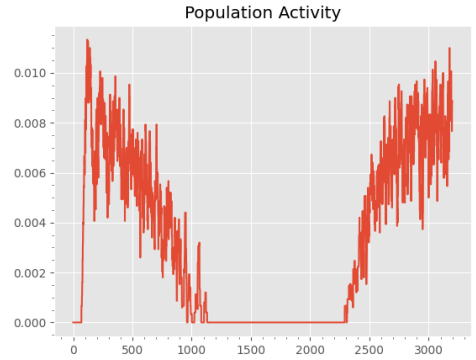

Voltage Of A Neuron
−40
−50
−60
−70
0
20
40
60
80
100

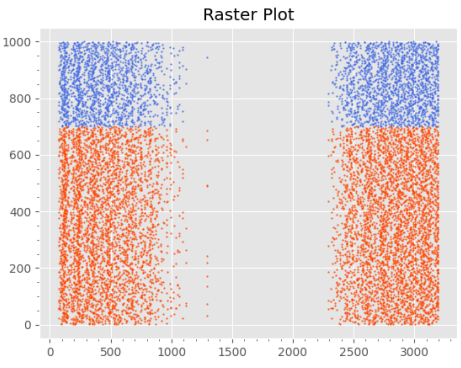

Voltage Of A Neuron
−40
−50
−60
−70
−80
0
20
40
60
80
100

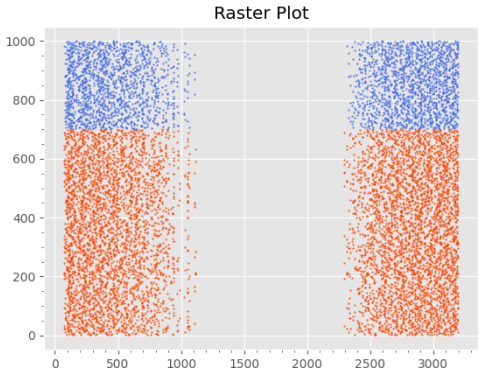
And as we would expect the 2nd set is more active and also what has happened here is that around our 1500th index we don't have an External input but we're seeing spikes which is caused by synapses

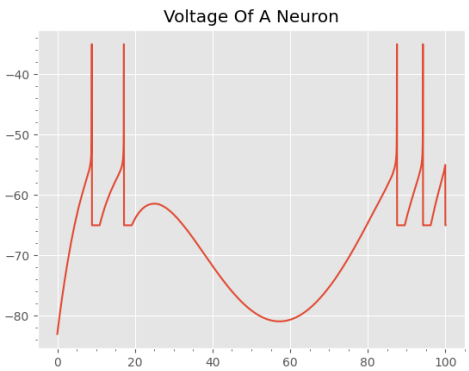
Both of the columns above we're fully connected populations, here we see the population with connection chance of 0.3 :

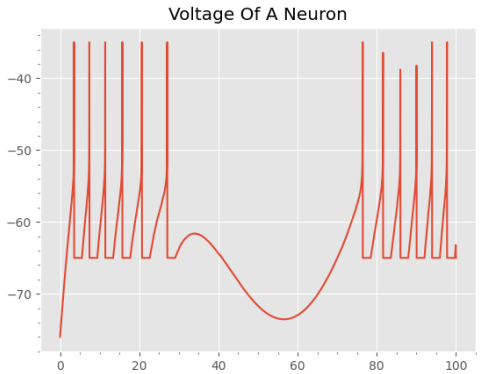












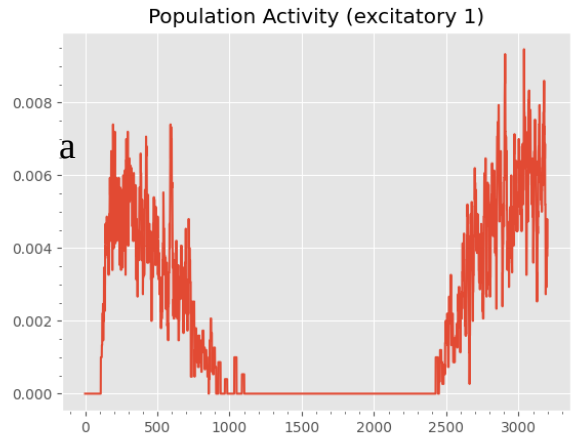
Part 2 :

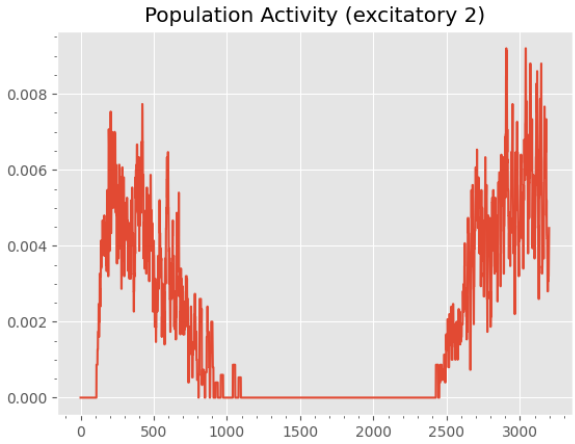
3 populations (2 excitatory and 1 inhibitoy)

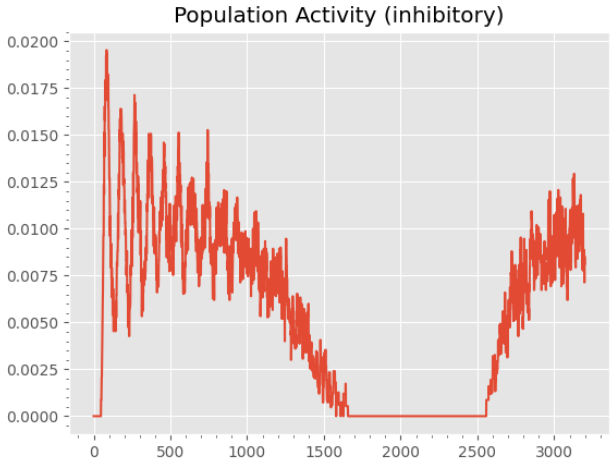
For the first set we use the same External input as we did before for the both excitatory population and for that we see the same results for both :

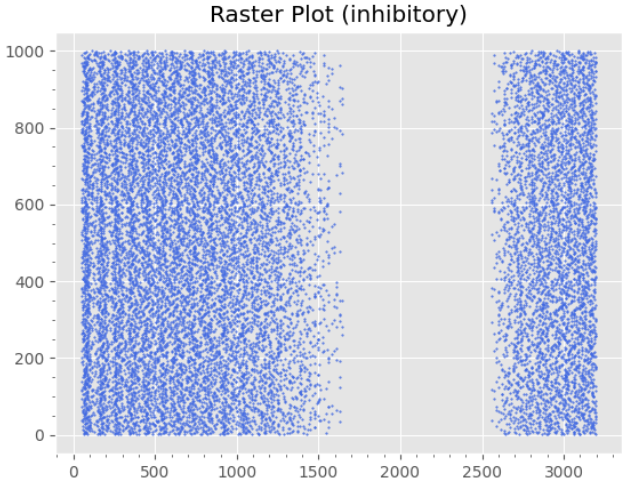
sa

a

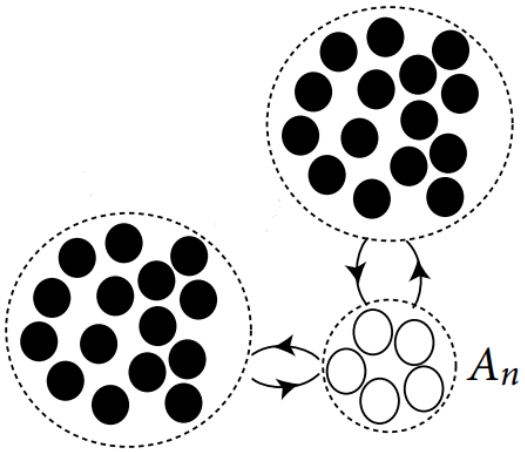








The 3 poplulations are connected together by this diagram :



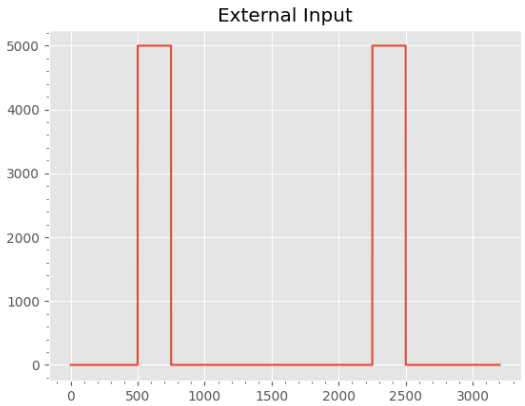
W13 and W23 are the inputs to our inhibitory population which is generated by the spikes of excitatory populations (meaning that when the excitatory populations start to spike, our inhibitory population which has no External input, but uses the Internal input of excitatory populations as an External inut, and gets active when the other 2 gets active, in order to reduce their activity by W31 and W32 which becomes a negative term applied to all the neurons of excitatory populations

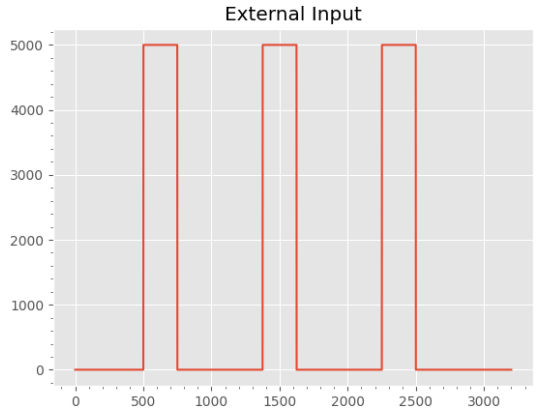
Here's the code of how that happens :

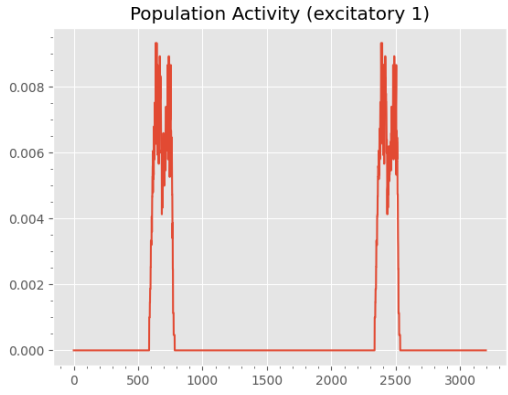
```
for i in range(1, 3200) :
    population_ex1.Next_tick(i)
    population_ex1.Adjust_current(i)
    for neuron in population_in.neurons :
        neuron.I[i+1] -= w31 * population_ex1.I_in[i]

    population_ex2.Next_tick(i)
    population_ex2.Adjust_current(i)
    for neuron in population_in.neurons :
        neuron.I[i+1] -= w32 * population_ex1.I_in[i]
```

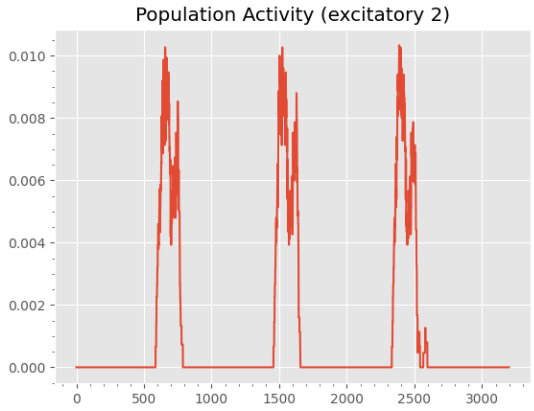
I tested different values for W13 and W23 and W31 and W32,W13 and W23 had to be generally higher amounts as their meant to produce enough Current for our inhibitory population to start spiking

For another example I used different External inputs for each of our excitatory populations, here's the results :


External Input
5000
4000
3000
2000
1000
0
0
500
1000
1500
2000
2500
3000

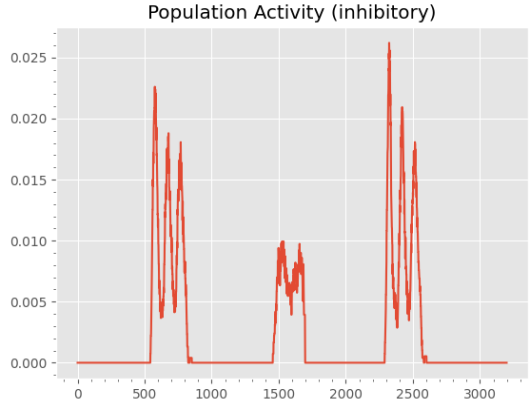

External Input
5000
4000
3000
2000
1000
0
0
500
1000
1500
2000
2500
3000


Population Activity (excitatory 1)
0.008
0.006
0.004
0.002
0.000
0
500
1000
1500
2000
2500
3000


Population Activity (excitatory 2)
0.010
0.008
0.006
0.004
0.002
0.000
0
500
1000
1500
2000
2500
3000


Raster Plot (excitatory 1)
1000
800
600
400
200
0
500
750
1000
1250
1500
1750
2000
2250
2500


Raster Plot (excitatory 2)
1000
800
600
400
200
0
500
750
1000
1250
1500
1750
2000
2250
2500


Population Activity (inhibitory)
0.025
0.020
0.015
0.010
0.005
0.000
0
500
1000
1500
2000
2500
3000


Raster Plot (inhibitory)
1000
800
600
400
200
0
500
750
1000
1250
1500
1750
2000
2250
2500