UNIVERITY OF TEHRAN

NEURAL POPULATION AND DECISION MAKING

IMPLEMENTATION OF SPIKING NEURAL POPULATION MODELS, DYNAMICS OF DECISION MAKING AND THEIR BEHAVIORS ANALYSIS

AUTHOR

ARASH NIKZAD

Student No. 610301195

SUPERVISOR

MOHAMMAD GANJTABESH

TEHRAN, MARCH 2024

Contents

Contents i

1 Introduction 2

2 Mechanism of synapses 3
2.1 Dirac delta-based synaptic mechanism . . . 3
2.1.1 LIF as building blocks . . . 3
2.1.2 RAELIF as building blocks . . . 4
2.2 Conductance-based synaptic mechanism . . . 5

3 Connectivity scheme 7
3.1 Full connectivity . . . 7
3.2 Random connectivity: fixed coupling probability . . . 9
3.3 Random connectivity: fixed number of presynaptic partners . . . 11
3.4 Noise sensitivity . . . 12
3.4.1 Full connectivity . . . 12
3.4.2 Random : fixed coupling probability . . . 13
3.4.3 Random : fixed number of presynaptic partners . . . 14

4 Balanced networks 15
4.1 Excitatory-Inhibitory population . . . 15
4.2 Synaptic Weights . . . 17
4.3 In-depth analysis . . . 18
4.3.1 Network Scale . . . 18
4.3.2 Building block . . . 20

5 Decision making 21
5.1 Simulation . . . 21
5.2 Change of mind . . . 22
5.3 Generalized winner-takes-all network . . . 23
5.4 In-depth analysis . . . 26
5.4.1 Synaptic weights . . . 26
5.4.2 Synaptic connectivity . . . 28

6 Conclusion **30**

1

Introduction

In the previous report, we implemented our first single spiking neuron models using one of the most basic and yet powerful models known as **LIF** and its variations. Now that we have gathered a sufficent understanding of our brain in a micro level, it is time to take our analysis into the macro level, **neural populations**. As we know, if you want to give only one reason behind the power of neural networks and computational models in their task and even the brain itself, is the power of these single neurons as a population. We saw that a single neuron can generate spikes in different patterns but a single neuron does not posses nearly enough power to be used in real life *intelligent activies*, so, in this report we try to simulate functionality of a neural population and synaptic connections, working with inhibitory and excitatory neurons to make a balanced population and at the end we focus on a model of decision making and we analysis their behavior in each step.

2

Mechanism of synapses

In this section, we introduce the dynamics of a synaptic connectivity and have our first neural population using two different methods of **conductance-based** and **dirac delta-based** synapses.

2.1 Dirac delta-based synaptic mechanism

We study a large and homogeneous population of neurons. In this synaptic mechanism, we focus on the the effects of pre-synaptic neuron spikes with an initial constant synaptic weight of w_{ij} that is independant of neuron's conductance at every moment. This helps to have a steady and somewhat predictable behavior for our population. The dynamics of this synaptic mechanism is as follows.

Dirac delta-based synaptic mechanism

$$I_i = \sum_{j=1}^{N} \sum_{f} w_{ij} \delta(t - t_j^{(f)}) + I^{ext}(t) \tag{2.1}$$

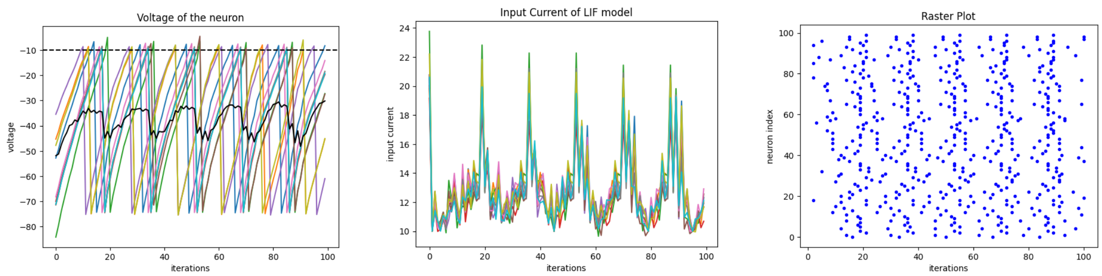
$I^{ext}(t)$ denotes the external input current coming into each neuron and w_{ij} is the synaptic weight between pre-synaptic neuron j and post-synaptic neuron i.

This mechanism simply simulates a weighted sum over the pre-synaptic neurons that fired a spike at time t changing the input current of post-synaptic neuron i. Now we can see the behavior of our first neural population, and try to see its behavior on different input currents and single spiking neuron models as the main building block.

2.1.1 LIF as building blocks

Here we used LIF as the building block of the population with `size = 100` for hundred iterations with a constant input of 10mA without noise.

As we can see the population works just fine, in fact one important thing is that, despite the fact that the external input current did not have any noise, we do not see a



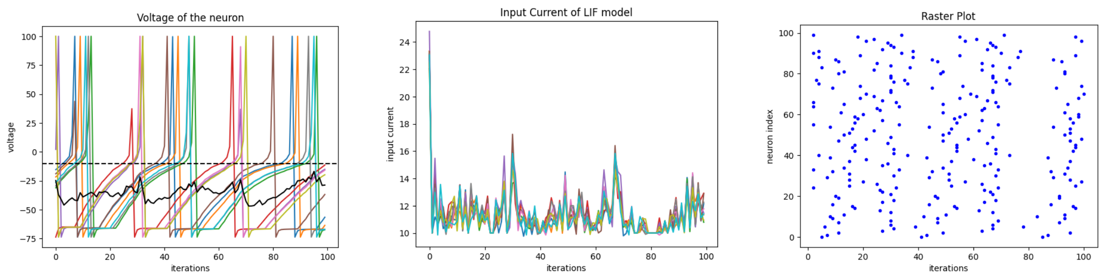
(a) *potential of first ten neurons in the population* **(b)** *input current of first ten neurons in the population* **(c)** *raster plot of all neurons in the population*

Figure 2.1: *dirac delta-based population with LIF*

steady input, the reason behind this, is the fact now, in a population, neuron's spike firing effect each others input current resulting in a different input current for each neuron with fluctuation. Now we can use a more advanced model we developed in the previous project, and seeing the difference in the activities of the population with more advanced building block.

2.1.2 RAELIF as building blocks

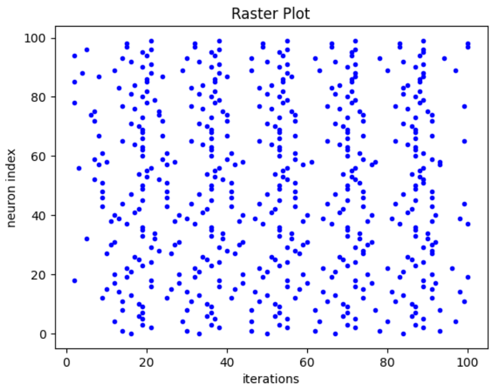
This model is the final model that we developed having the most complex dynamics out of all of the previous single spiking neuron models. As we know, behaviors of this model is considerably different to our base LIF model, here we can test, whether that difference in building blocks makes that much of a difference in a neural population or generally in macro level.



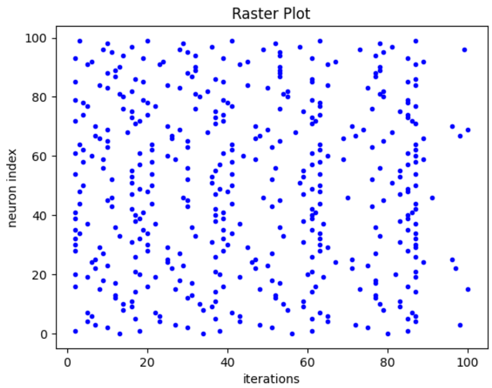
(a) *potential of first ten neurons in the population* **(b)** *input current of first ten neurons in the population* **(c)** *raster plot of all neurons in the population*

Figure 2.2: *dirac delta-based population with RAELIF*

Well, there are differences for sure and it is not something unexpected. The RAELIF model has two main feature which makes its population differ considerably with our previous LIF-based population in Figure 2.1. The first one is the **adaptability** and the second yet more effective one is **refractory period**. With both of these, not only the potential of each neuon is different but more importantly as we can see from the raster plot the population activity has decreased as well. In macro level the potential of each neuron is not a matter of high importance for most tasks and the accumulative results which can be shown as **population activity** or **raster plot** of the population, but what if we remove or lower the refractory period and only see the raster plot of these populations and compare them to each other.



(a) *LIF-based population raster plot*



(b) *RAELIF-based population raster plot*

Figure 2.3: *LIF-based vs. RAELIF-based populations*

As we can see, despite of the fact that RAELIF model has more terms and is more complex in its dynamics, we results might not be that much different after all, of course it is observable that as the iterations passes the density of taster plot in RAELIF model decreases due to its **adaptability** but one thing to consider is not only we do not always need our models to be adaptive, but also the computation time of running these models for `iter = 100` has a huge gap, RAELIF model runs in `119.72ms` while LIF model runs in `70.85ms` which is a considerable percentage while not having a huge difference so in most of cases where the particular activity of a single neuron is not that important and only the spiking and activity as a population is a matter on importance we would prefer LIF over heavier models.

2.2 Conductance-based synaptic mechanism

This mechanism is more developed in comparison to the previous one. Using conductance-base synapses not only the spikes and activities of other neurons in the population effect the input current of each neuron but also it has a control over the input current entering the neuron due to spike of a pre-synaptic neuron, with accordance to its potential. using a new dynamics to change the potential of each synapse $g_{syn}(t)$ gives a more dynamic to the population instead of somewhat static synaptic weights.

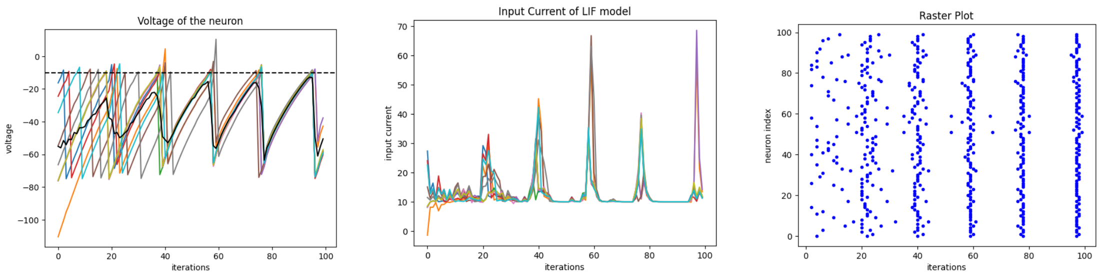
Conductance-based synaptic mechanism

$$\frac{dg_{syn}(t)}{dt} = \bar{g_{syn}} \sum_k \delta(t - t_k) - g_{syn}/\tau_{syn} \tag{2.2}$$

$$I_{syn}(t) = g_{syn}(t)(V(t) - E_{syn}) \tag{2.3}$$

$\bar{g_{syn}}$ denotes the maximum conductance elicited by each spike and E_{syn} is the reversal potential.

Here we only use LIF model as our building block as we explained in the previous section that using high-level generalized version of LIF might not make much of a difference after all, but we can always test out those single neuron models when in need.



(a) *potential of first ten neurons in the population* **(b)** *input current of first ten neurons in the population* **(c)** *raster plot of all neurons in the population*

Figure 2.4: *conductance-based population with LIF*

Here we can see the difference in dirac-delta and conductance based populations. The most important factor is the **synchronous** in this population comparing to the dirac delta based one. We can see that after a passage of time, neuron's spikes become in sync meaning that the more time goes on, despite the fact that they start with different initial potential and due to that they spike at different times, the time of spikes happen in shorter range of time making them in sync. The reason behind this is the ability we just gave to our network. Because the synaptic weights are accumulated based off of their potential as the network goes on and random spikes happen this has more effect on the ones with higher potential making them spike faster than usual, on the other hand, it has lower effect on the ones with lower potential making them wait longer for the next spike. This phenomenon make population in sync after a period of time. of course here we only used excitatory neurons which is not the case in a real neural network.

Bear in mind that not only the synaptic mechanism matters in a population but also their connectivity scheme, in the next chapter we are going to implement and see the behavior of different connectivity scheme in a neural population.

3 Connectivity scheme

In Chapter 2, we made our first neural population with an interaction between its neurons called synapse. We can look at a neural population or as one would say a neural network as a graph. A graph has two main components, **vertices** as neurons and **edges** as synapses. In the previous chapter the graph that we were working on was a compelete weighted graph, however even in our brain on a biologcal point of view not all neurons are effected by all other neurons in a network making in so dense and having not much room for diversity of behaviors. Due to these facts we have different connectivity schemes to choose from based on the needs of the population and the goal we try to achieve.

3.1 Full connectivity

This is the most basic one which is somewhat the thing we used so far as the base of our population. the idea is to have a compelete graph, having each neuron have both pre and post synapses to any other neuron in the network.

all-to-all connectivity within a population. having somewhat same strength for each w_{ij}.

$$w_{ij} = \frac{J_0}{N} \tag{3.1}$$

or

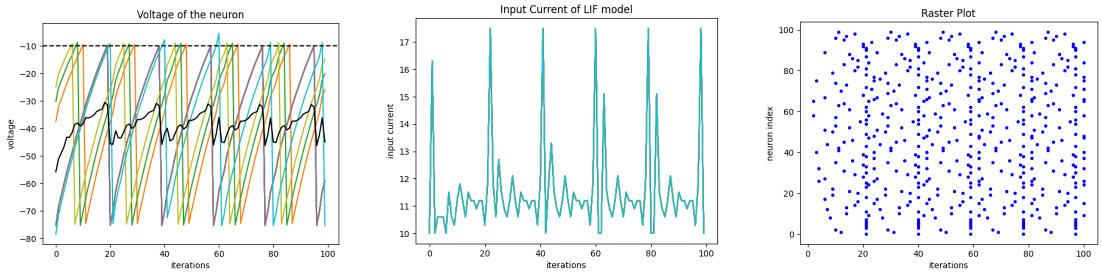
$$w_{ij} = normal(\frac{J_0}{N}, \frac{\sigma_0}{N}) \tag{3.2}$$

where J_0 is a constant number and N is the size of the population.

We can see the effects of this connectivity scheme with LIF model as building blocks and dirac delta-based synaptic weights.

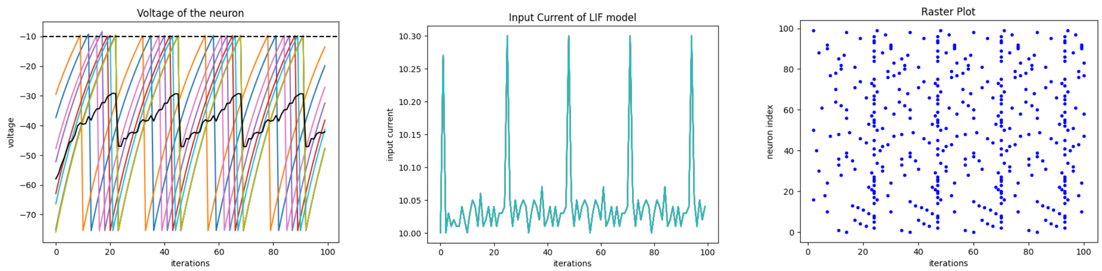
In Figure 3.1, we can see that due to the fixed weights there are not much diversity in spikes and the initial differences in potential is the basic difference in their behavior. now let's see the effects of J_0.

Here in Figure 3.2, it is obivous that it seems like there is no population, and their connection does not matter because the weight of these synaptic connection are really



(a) *potential of first ten neurons in the population*
(b) *input current of first ten neurons in the population*
(c) *raster plot of all neurons in the population*

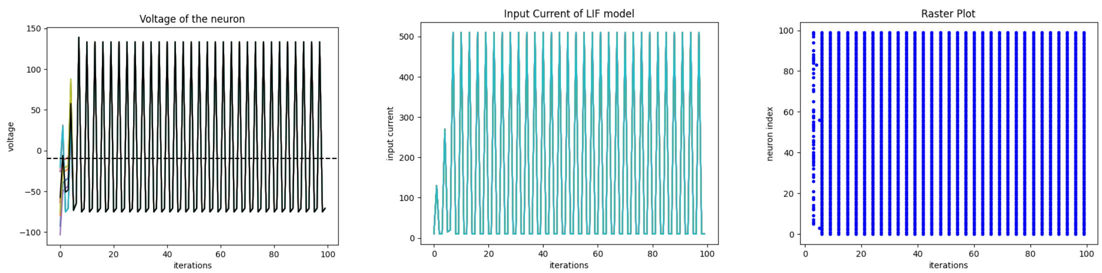
Figure 3.1: *full connectivity scheme with* $J_0 = 50, N = 100$



(a) *potential of first ten neurons in the population*
(b) *input current of first ten neurons in the population*
(c) *raster plot of all neurons in the population*

Figure 3.2: *full connectivity scheme with* $J_0 = 1, N = 100$

weak that do not play any role in their potential and also in networks activities.



(a) *potential of first ten neurons in the population*
(b) *input current of first ten neurons in the population*
(c) *raster plot of all neurons in the population*

Figure 3.3: *full connectivity scheme with* $J_0 = 100, N = 100$

Now Figure 3.3 shows that if we set J_0 too high, the neural population functionality breaks, meaning that all neurons will be active for each iteration without even an incoming external input making it have the same behavior all over again which is not useful. On the matter hand in this method not the density makes the computation rather heavy but also due to the fact that J_0 is independant of population size, by chaning the size of a network it can go out of balance which is not what we want, so there must be other connectivity schemes to solve these problems.

3.2 Random connectivity: fixed coupling probability

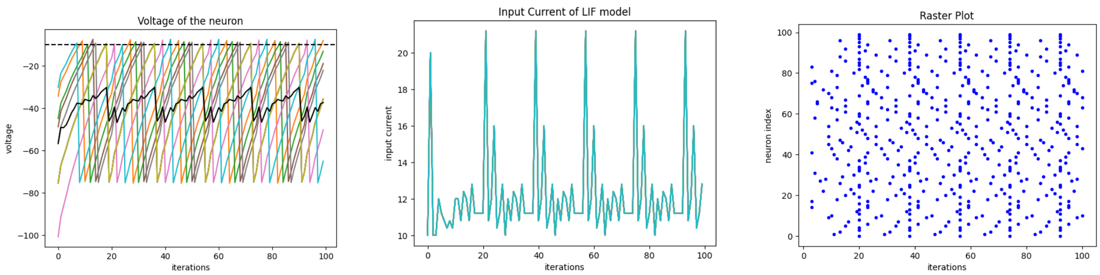
In this connectivity scheme, not all neurons are conncted, in fact we have a sense of control over these connections. In this scheme as the name suggests, we choose connections randomly with a probability p among all the possible N^2 connections. This helps to add more diversity to the population which solves one of the problems of the first method.

$$mean(C) = pN \quad Var(C) = p(1-p)N \tag{3.3}$$

$$w_{ij} = \frac{J_0}{C} = \frac{J_0}{pN} \tag{3.4}$$

where C_i is number of pre-synaptic connections to a post-synaptic neuron i.

Now we can the effects of this scheme with different values of p and with $J_0 = 40$ for all synapses.



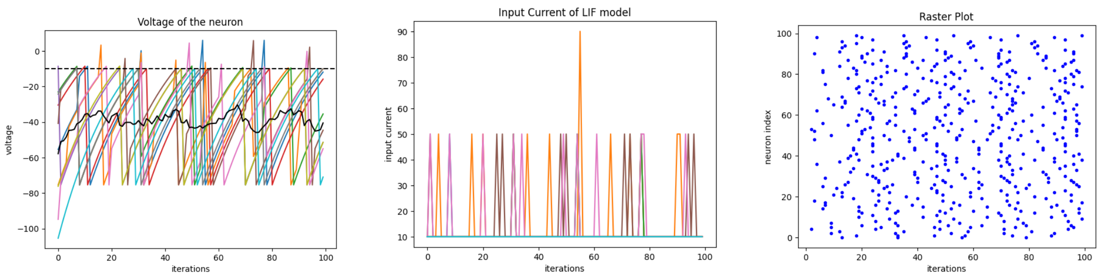
(a) *potential of first ten neurons in the population* **(b)** *input current of first ten neurons in the population* **(c)** *raster plot of all neurons in the population*

Figure 3.4: *random connectivity scheme with* $p = 1$

Here we can see that because $p = 1$, it is basically a full connectivity scheme, which looks just as the same. The power of this scheme lies in using a right value of p, setting in too high would result in a much dense network and acts similarly to full connectivity scheme and as you can see all of the input current match exactly. now let's try the other end, using very low value for p.

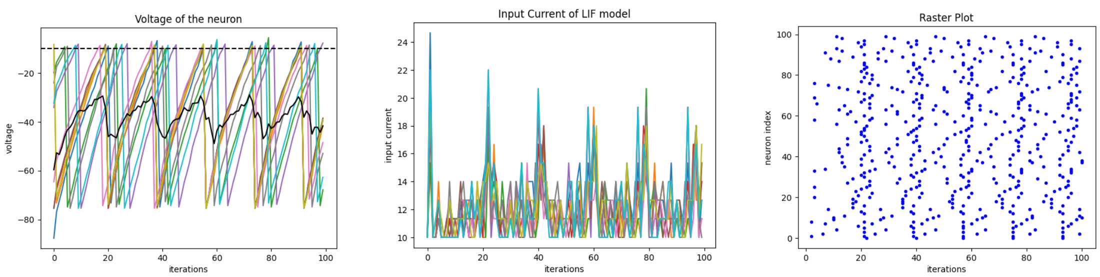
here we can see that there are in fact difference in their input currents but its ready subtle and because the connections are hardly made for each neuron there is just like a spike of input current and goes back to normal also because the synaptic weights are dependant on p, the weights are considerably high which is not a desirable result. Now that we have seen both ends, we can use a somewhat logical value for p to see the difference.

Here is mostly what we wanted. Now there are so subtle or abrupt changes in input current and the diversity of the population is reserved which is a great accomplishment, on the other hand due to the decrease in connections less computation power is needed



(a) *potential of first ten neurons in the population* **(b)** *input current of first ten neurons in the population* **(c)** *raster plot of all neurons in the population*

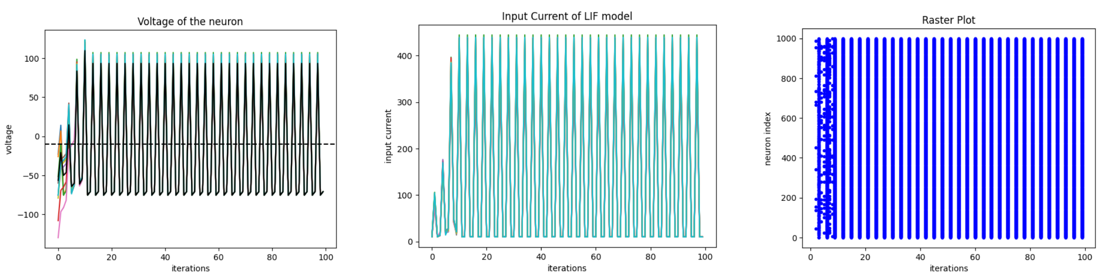
Figure 3.5: *random connectivity scheme with* $p = 0.01$



(a) *potential of first ten neurons in the population* **(b)** *input current of first ten neurons in the population* **(c)** *raster plot of all neurons in the population*

Figure 3.6: *random connectivity scheme with* $p = 0.3$

for the same result. the only problem we have is that this population is sensitive to **scaling**, meaning that for a population of 100 neurons around 30 connections is logical and the accumulative generated input current for the post-synaptic neuron is logical, however if you scale the network into 10000 neurons for instance, then, the number of connections for each neuron would be really high resulting in malfunctioning of the network as we saw in Figure 3.3. If we can solve this problem while still maintaining the diversity and low computation cost, we have reached a desirable connectivity for most cases.



(a) *potential of first ten neurons in the population* **(b)** *input current of first ten neurons in the population* **(c)** *raster plot of all neurons in the population*

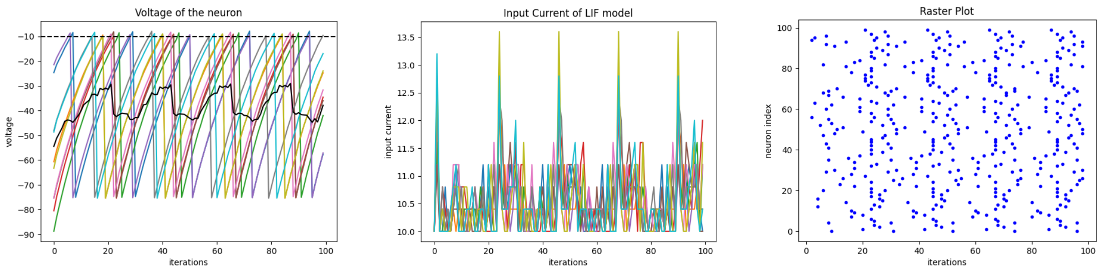
Figure 3.7: *random connectivity scheme with* $p = 0.3$ *of population of* 1000 ***neurons***

In Figure 3.7, it is obivous that while the same setting works just fine in a network of

100 neurons, malfunctions in scale of 1000 neurons.

3.3 Random connectivity: fixed number of presynaptic partners

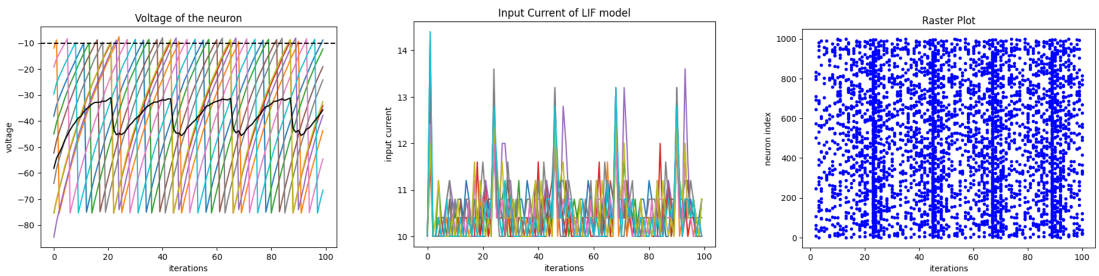
As we saw in the previous section, a fixed connection probability cannot be correct in simulations, on top of that, not all neurons are and should be active on the same time, so in order to fix all of our problems we can use a scheme in which the number of pre-synaptic connection C is fixed and only the subset of neurons in which a neuron i should be conncted is chosen randomly but not its size, making it both have diversity and low computation cost, as well as not having the scaling problem.



(a) *potential of first ten neurons in the population* **(b)** *input current of first ten neurons in the population* **(c)** *raster plot of all neurons in the population*

Figure 3.8: *random connectivity scheme with* $C = 30$ *of population of* 100 ***neurons***

Here we can see that with fixed random connections of $C = 30$, the population works just fine with good diversity and low computation cost, now it is time to see its behavior while facing the **scaling problem**.



(a) *potential of first ten neurons in the population* **(b)** *input current of first ten neurons in the population* **(c)** *raster plot of all neurons in the population*

Figure 3.9: *random connectivity scheme with* $C = 30$ *of population of* 1000 ***neurons***

As it seems in Figure 3.9, this connectivity scheme has overcome all obsticles including scaling, it looks really close to the neuron population behaviors of 100 neurons as we wanted it to be. The only difference might be in the raster plot which feels a lot denser, however the only reason behind its density is that it shows the spike pattern of 1000 neurons in the scale plot scale, for better understanding of how the main behavior of the population has remained the same, we can take a look at the population activity plot of both networks.

population activity

$$A(t) = \lim_{\Delta t \to 0} \frac{1}{\Delta t} \frac{n_{act}(t; t + \Delta t)}{N} = \frac{1}{N} \sum_{j=1}^{N} \sum_{f} \delta(t - t_j^f) \tag{3.5}$$

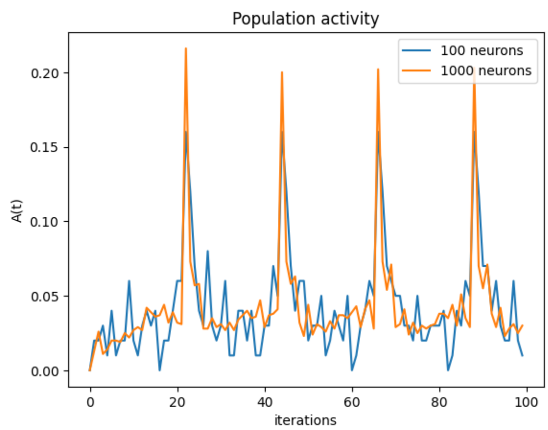


Figure 3.10: *population activity of neurons with 100 and 1000 neurons*

Here it is clear that the difference is minimum,essentially, having the same population activity, which is an indicator of the fact that we have resolved the scaling problem with this connectivity scheme fixing the main problem we have the basics of neural population.

3.4 Noise sensitivity

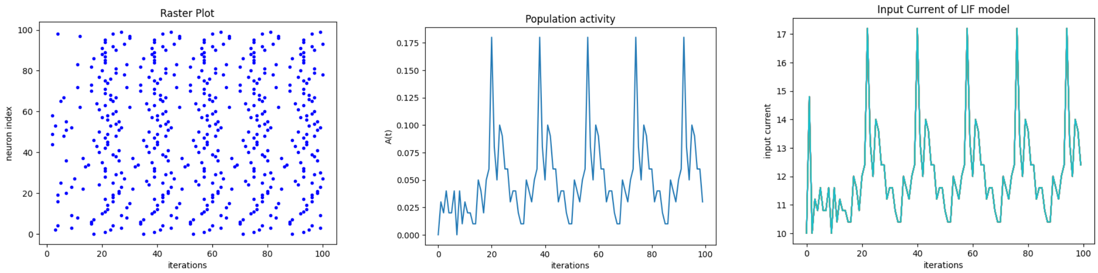
The final section of this chapter is dedicated to comparing the three connectivity schemes that we implemented in their **noise sensitivity**. In order to see their sensitivity, we first use a constant external input current without considerable noise to the network as we did in the previous sections, after that we compare the results to external input current of heaviy noise and see if the network is highly effected by noisel.

3.4.1 Full connectivity

Here we start with the first connectivity scheme, the full connectivity. Due to the fact that all neurons are conncted to each other we expected to see a considerable change in the behavior of activites and input currents of the population.

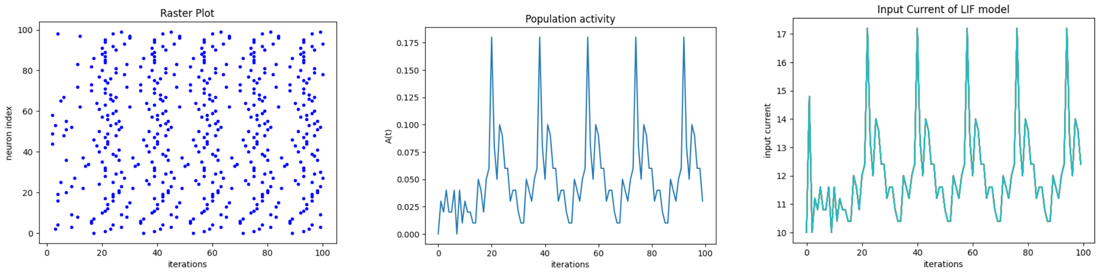
As we can see the changes are clear. The effecs of heavy noise has a direct effect on not only each neuron but also the whole network. In order to have a more vivid look to the changes we can plot the results of both population activity and input currents together.

The results are completely obivous in these plots. Figure 3.13 shows that the heavy noise is highly effective on the results of the population and the population is **highly**



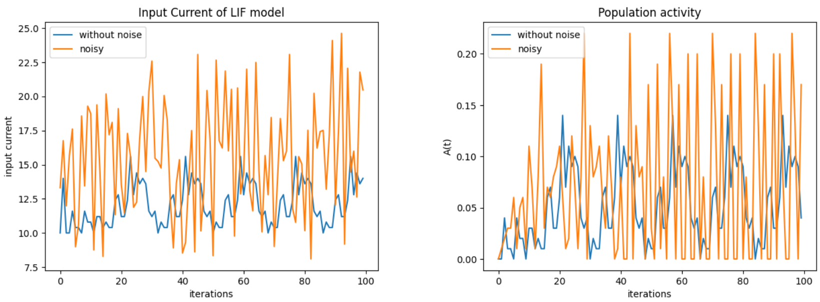
(a) *raster plot of neurons in the network* **(b)** *population activity neurons in the network* **(c)** *input current of neurons in the network*

Figure 3.11: *population with external input current of no noise (full connectivity)*



(a) *raster plot of neurons in the network* **(b)** *population activity neurons in the network* **(c)** *input current of neurons in the network*

Figure 3.12: *population with external input current of noise (full connectivity)*



(a) *input currents* **(b)** *population activities*

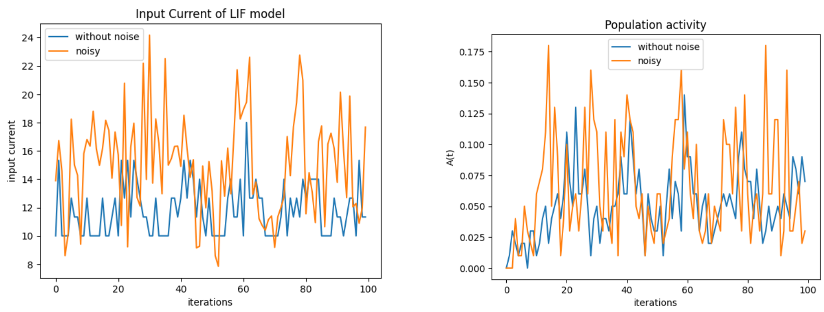
Figure 3.13: *comparison between noisy and without noise external input current (full connectivity)*

sensitive to noisy input.

3.4.2 Random : fixed coupling probability

This connectivity scheme is a bit different to the previous one as we saw, due to its random nature we expect to see a lower level of effectiveness of noisy input to the population. For a more convenient understanding of the differences we only use the population activity and their input current to compare the two results of the population with and without noise in their input.

As we can see in Figure 3.14, Although the noisy input has a drastic effect on the input current due to the heavy noise in their external input, The population activity of



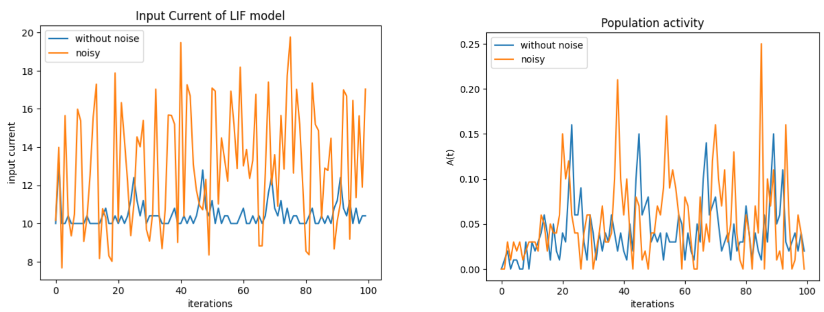
(a) *mean input currents* **(b)** *population activities*

Figure 3.14: *comparison between noisy and without noise external input current (Random : fixed coupling probability)*

these network are much closer in comparison to **full connectivity** as expected. So the *random fixed coupling probability scheme* is **relatively less sensitive** to noisy input which is another advantage of these networks in most cases, of course we we want to have a highly sensitive neural population (for instance for pain sensors), the networks with full connectivity might be a better option.

3.4.3 Random : fixed number of presynaptic partners

We have seen that these networks generally do well when facing different challenges like scaling problem and heavy synaptic weights so it would not be surprising to see that this connectivity scheme is less sensitive to noisy input than full connectivity but we have to see the results in order to be able to compare it to the previous random connectivity scheme.



(a) *mean input currents* **(b)** *population activities*

Figure 3.15: *comparison between noisy and without noise external input current (Random : fixed number of presynaptic partners)*

Here in Figure 3.15 we can clearly see that this scheme is less sensitive in comparison to full connectivity however, Both of random connectivity schemes perfom relatively well to noisy input, but none of them out-perform the other in this topic.

Now that we have assembled the basic tools to simulate a population it is time to offically introduce **excitatory** and **inhibitory** populations and try to simulate our first balanced network.

4 Balanced networks

In our brain as a neural network, not all neurons has positive effect on their potential and spiking of other neurons or in another words not all neurons are **excitatory**. There are many reasons behind this phenomenon, but if we want to address a few, one of the reasons is the fact that activites of different set of neurons might be in contrast to activities of another set of neurons, so these populations not only should have excitatory connections but also they have to inhibit the other to spike this mechnism is disccused in further details in Chapter 5. Another reason is what we saw in Chapter 3, if all the connections of neurons are excitatory, then the accumulative result would cause the system to go out of balance, making it malfunction so there must be an **inhibitory** population to balance out the excitation and inhibition.

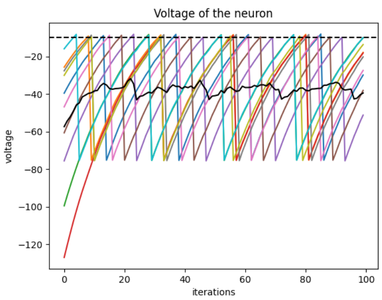
In this chapter, we focus on making our first balanced network consisting of one **excitatory** and one **inhibitory** population.

4.1 Excitatory-Inhibitory population

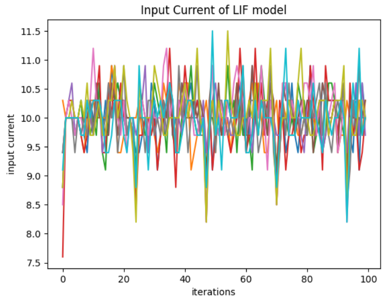
We first try to make two identical populations both in size and in their building blocks (both using LIF), also the connectivity scheme is at first set to be **random coupling connectivity** and analyse the behavior of such network.

There are things to attend about Figure 4.1. We can see from the input currents that this time, not all the changes are above 10 which is the base external input current, so there is some form of inhibition in our network, we can see that both excitatory and inhibitory population have somewhat but not exactly the same activity and spiking pattern. These behaviors are expected because all their connections and weightings are all same, but mostly the weights of inhibitory synapses are higher comparing to the excitatory one, to let the $J_0 = 200$ for inhibitory neurons and $J_0 = 30$ for excitatory ones without the inhibition for themselves and see how the network reacts to these adjustments.

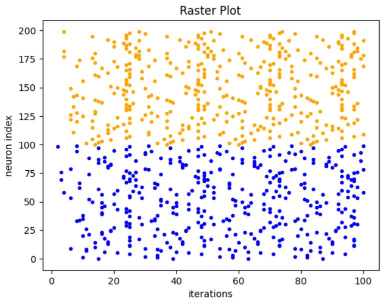
In Figure 4.2, the inhibition is clear, not only we can see that the population activity of the excitatory population has decreased but also the raster plot shows that there are less spikes for excitatory population.



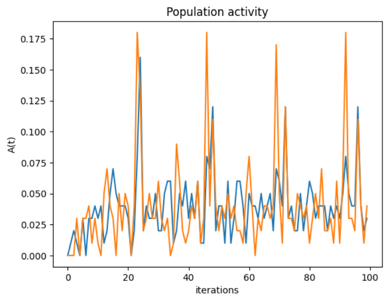
(a) *potential of first ten neurons in the network*



(b) *current of first ten neurons in the network*

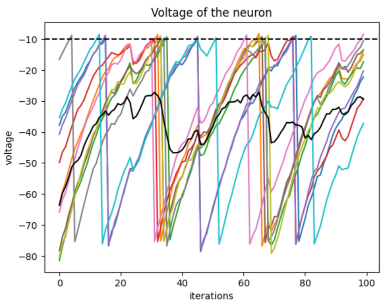


(c) *raster plot of the network*

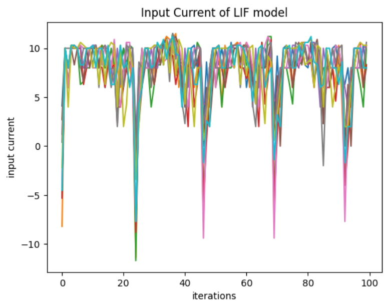


(d) *population activity of the network*

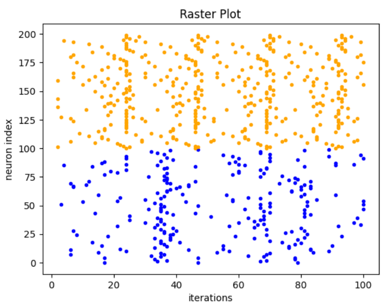
Figure 4.1: *exc-inh network of 100 neurons each*



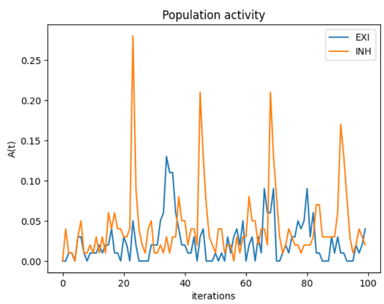
(a) *potential of first ten neurons in the network*



(b) *current of first ten neurons in the network*



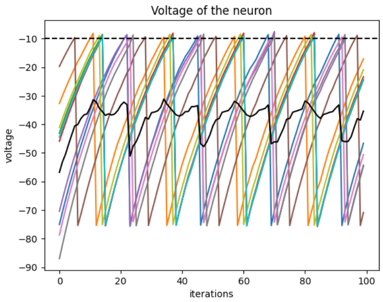
(c) *raster plot of the network*



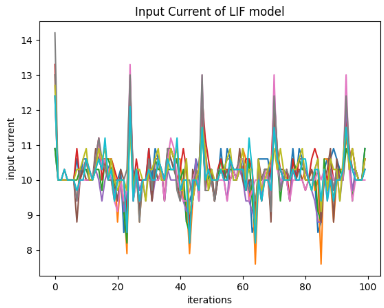
(d) *population activity of the network*

Figure 4.2: *exc-inh network of 100 neurons each*

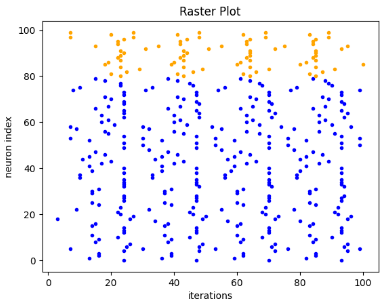
In out brain, the distribution of excitatory and inhibitory neurons is not equal. The studies on human brain show that around **80%** of neurons are excitatory and only around **20%** of them are inhibitory. So it is better to simulate the network as such.



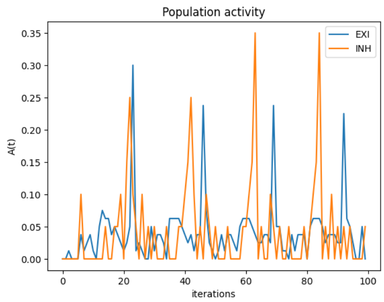
(a) *potential of first ten neurons in the network*



(b) *current of first ten neurons in the network*



(c) *raster plot of the network*



(d) *population activity of the network*

Figure 4.3: *exc-inh network of 100 neurons (80/20)*

Now we can see that it is somewhat back to normal with $J_0 = 60$ for inhibitory and $J_0 = 30$ for excitatory ones. But using this connectivity scheme although keeps the network working but it is not always exactly balanced.

4.2 Synaptic Weights

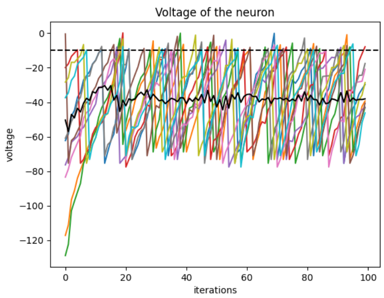
The studies have shown that in a balanced network consisting one excitatory and one inhibitory population, it is possible to adjust parameters such that the total amount of excitation and inhibition cancel each other, so that excitation and inhibition are balanced. we can scale Synaptic weights so as to control the amount of fluctuation of the input current around zero, on top of that, there is no need for rescaling scheme, the mean input would be around **zero** automatically using this weighting.

Balanced weighting

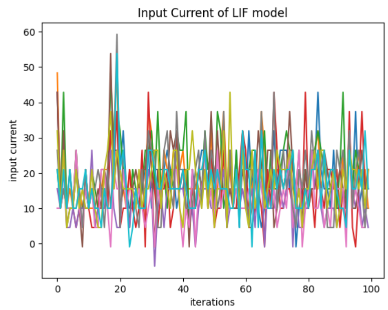
$$w_{ij} = \frac{J_0}{\sqrt{C}} = \frac{J_0}{\sqrt{pN}} \tag{4.1}$$

This weighting uses the fixed coupling probability or fixed number of partners scheme.

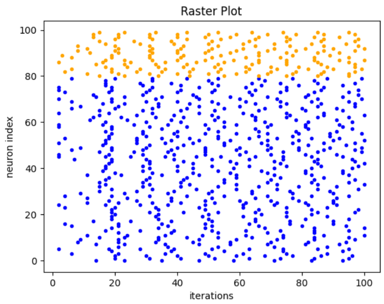
Now let's see the effects of balanced weights on a network of 80-20 excitatory and inhibitory neurons with same J_0 for all the network.



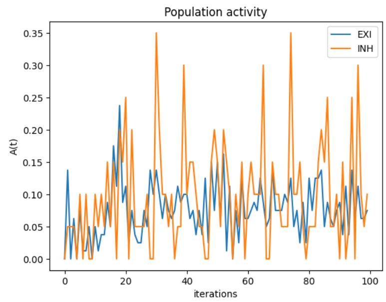
(a) *potential of first ten neurons in the network*



(b) *current of first ten neurons in the network*



(c) *raster plot of the network*



(d) *population activity of the network*

Figure 4.4: *exc-inh network of 100 neurons (80/20)*

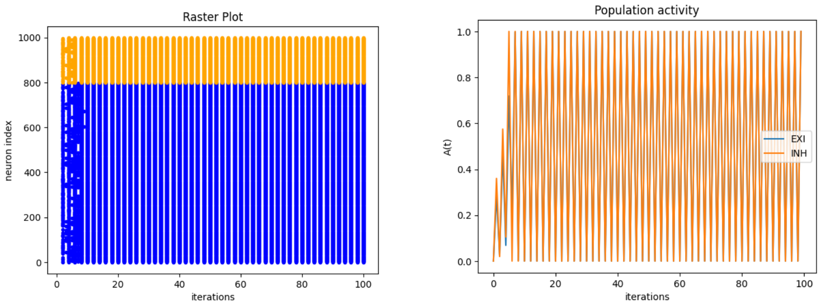
Here in Figure 4.4, we can vividly notice that in contrast to the previous ones that using heaver synaptic weights were needed for inhibitory population to keep the network in balance, there is no need here. The input current clearly shows that these currents are more intact and with less fluctuation comparing to the previous methods and we still have considerable population activity for both inhibitory and excitatory populations.

4.3 In-depth analysis

In this section, we are going to analyse the behavior of our network with different parameters and try to find the constraints of them so that the network is at balance. We have done some analyse regarding the J_0 value of the inhibitory or excitatory population and also the connectivity mechnism at a glance.

4.3.1 Network Scale

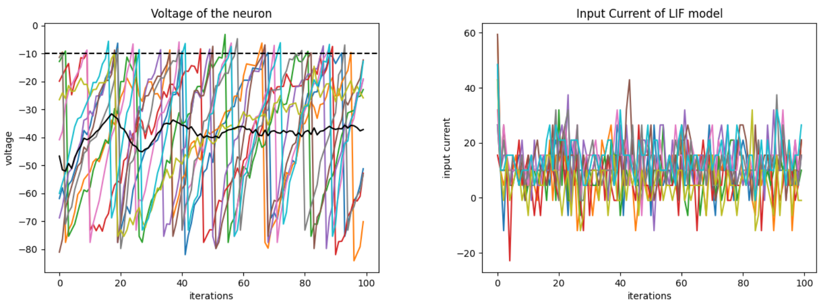
One important aspect of the network that we developed so far was the fact that it can handle the scaling problem, not as we use the **fixed coupling probability scheme**, we saw in Chapter 3 it cannot handle scaling properly. Here we want to test to see wether we encounter the same problem or not. Keep in mind that we still maintain the 80/20 percentage distribution over excitatory and inhibitory populations.



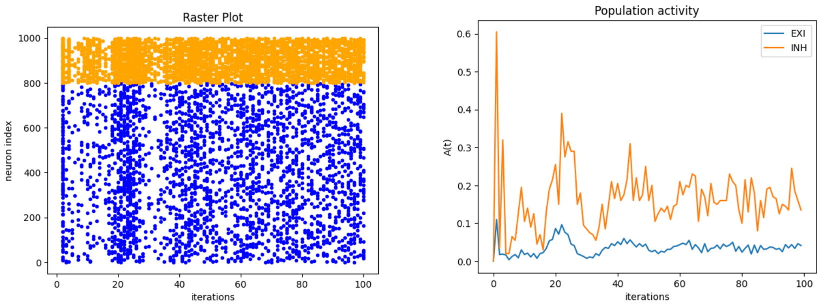
(a) *raster plot of the network* **(b)** *population activity of the network*

Figure 4.5: *exc-inh network of 1000 neurons (80/20) - fixed coupling*

Unfortunately, it does work better than usual, but even with this weighting the network is not immune to scaling and can go out of balanc easily, now we can try the network of balanced weights using the **fixed number of partners scheme**.



(a) *potential of first ten neurons in the network* **(b)** *current of first ten neurons in the network*



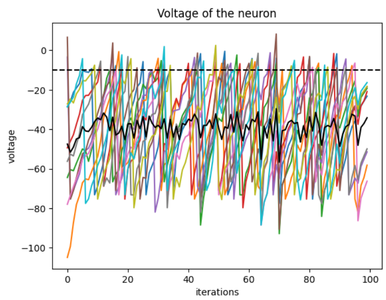
(c) *raster plot of the network* **(d)** *population activity of the network*

Figure 4.6: *exc-inh network of 1000 neurons (80/20) - fixed number of partners*

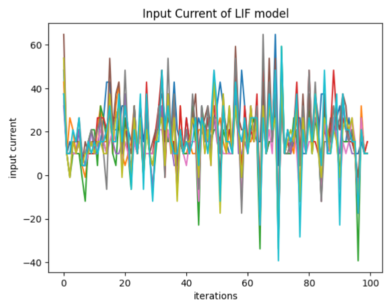
Figure 4.6 has the set of plots for a real balanced network, not only resistant to scaling but also the input current and the mean potential is more centered around their base values and in general the activities of the excitatory population has less fluctuation which is the desired outcome.

4.3.2 Building block

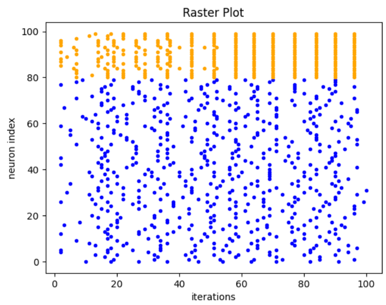
In biology, the studies have shown that excitatory neurons are more senstive. Due to their smaller size, they have to work faster and be more sensitive to input current that excitatory neurons (so-called **fast-spike**). In order to achieve this behavior in our simulated neural network, we can have different τ for excitatory and inhibitory populations, setting $\tau_{inh} < \tau_{exi}$ and see the behavior they produce.



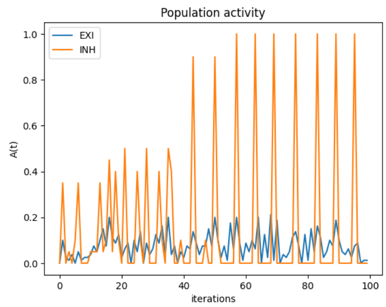
(a) *potential of first ten neurons in the network*



(b) *current of first ten neurons in the network*



(c) *raster plot of the network*



(d) *population activity of the network*

Figure 4.7: *exc-inh network of 100 neurons (80/20) -* $\tau_{exi} = 25ms, tau_{inh} = 20ms$

Now in Figure 4.7we can see that the fluctuation has increased in all aspects of the network however this this change we can see a clear effects of yet small but powerful inhibitory population. With this speed and power of inhibition the spike pattern of the excitatory neurons would change faster and gets into a stable phase which somewhat makes the decision making and final responses faster.

Now that we developed our first network we are ready to make use of it in a more practical way, as the first task of a neural network as general as **decision making**.

5

Decision making

Decision making is one of the most important abilities of the human mind, it serves lots of purposes and arguably is one of the most crucial tasks of our brains, so it begs the question. What do we consider **"decision making"**? In order to name a process, decision making it should have a few properties including, **input-output representation**, **selection process**, **feedback** and **measurement tool**. By these factors, one of the best options to focus on is **percueptual decision making**.

5.1 Simulation

One of the most basic ways we can simulate a decision making process, is using neural populations. Imagine we have only two options (A) or (B). and we want to have a model that based on the given input, it decides which one to choose. In order to make a competition between these two **excitatory populations** (A) and (B), we add an **inhibitory population** in between, this way with the activity of one side the other one gets inhibited to fire spikes and this goes back and force until one overcomes the other which we declare as the network's decision. This network is called **winnder-takes-all network**.

Dynamics of decision making (two decisions)

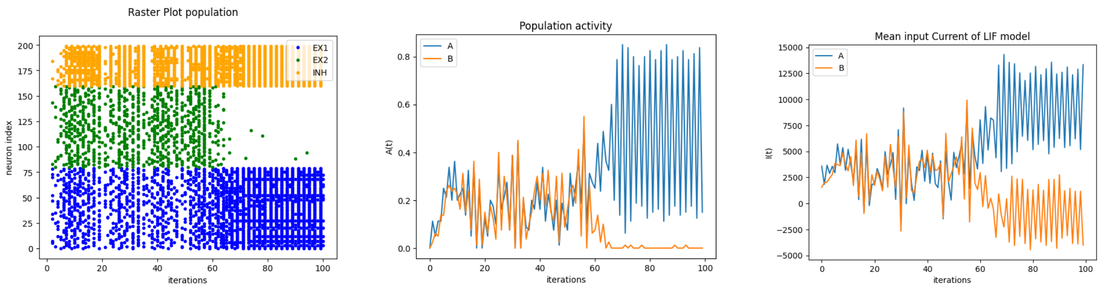
$$\tau_E \frac{d_{E,1}}{dt} = -h_{E,1} + w_{EE} g_E(h_{E,1}) + w_{EI} g_{inh}(h_{inh}) + RI_1 \tag{5.1}$$

$$\tau_E \frac{d_{E,2}}{dt} = -h_{E,2} + w_{EE} g_E(h_{E,1}) + w_{EI} g_{inh}(h_{inh}) + RI_2 \tag{5.2}$$

$$\tau_{inh} \frac{d_{inh}}{dt} = -h_{inh} + w_{IE} g_E(h_{E,1}) + w_{IE} g_E(h_{E,2}) \tag{5.3}$$

where g_E and g_{inh} are the gain functions.

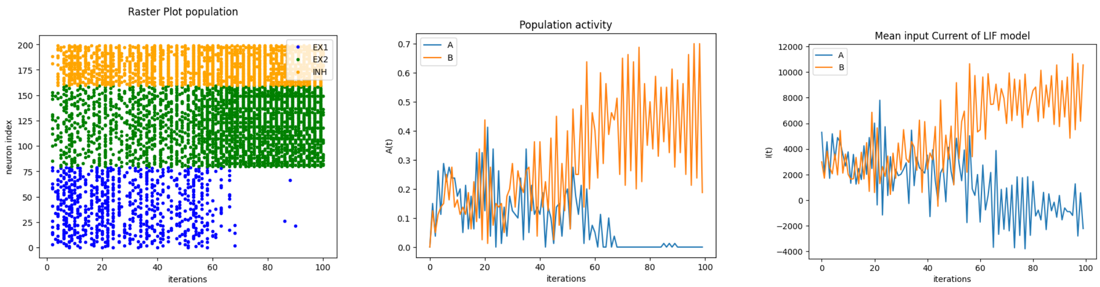
using this dynamics, we can have the first decision making network, first by having the same external input current, and after a while adding more to only one population (A) to see the results. (populations are of the sizes 80 for both excitatory populations (A) and (B), while 40 for the inhibitory population (80/20 percentage)).



(a) *raster plot of neurons in the network* **(b)** *population activity neurons in the network* **(c)** *mean input current of neurons in the network*

Figure 5.1: *Decision making network of 200 neurons*

In Figure 5.1, as we can see, at first both populations are active because they have the same external input current, however after the entering the more powerful input current to (A), we can see that its population activity increases while the activies of population (B) almost vanishes. This is vice versa, meaning if the same changes where applied to the external input current of (B), the winner of the decision would be different.



(a) *raster plot of neurons in the network* **(b)** *population activity neurons in the network* **(c)** *mean input current of neurons in the network*

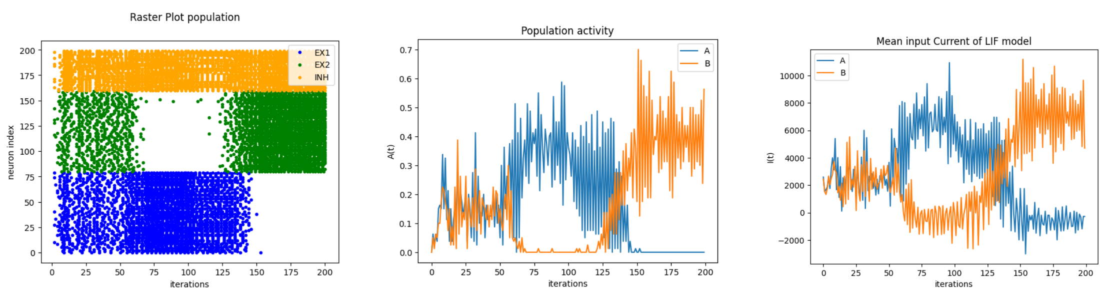
Figure 5.2: *Decision making network of 200 neurons*

Where as expected we can see that the table has turned. The change of the input current resulted in the change of decision of the network, which indicates that the decision making is not actually random and the process is dependant on the input current as we decided to be the input of our network.

5.2 Change of mind

from the previous section, we understood that our network has the ability of decision making between two choices using neural populations and variation of input current. Now what we want to test out, is the fact that if we swap the external input current, does that network change its decision or does it get stuck in the first one.

Fortunately, Figure 5.3, accepts our hypothesis, our model indeed can change its mind, as we can see at first it population activity of (A) increases while (B)'s decrease, on the other hand, ofter we swap the input current which is the would be the input,



(a) *raster plot of neurons in the network*

(b) *population activity neurons in the network*

(c) *mean input current of neurons in the network*

Figure 5.3: *Decision making network of 200 neurons*

networks knows that should decide (B) in the previous experiments, as a matter of fact, changes its mind and we can see their population activiy and raster plot exchanges making the new decision of the network (B). This is really powerful due to the fact that our model can be used online for a stream of inputs without having to be reset. Bear in mind that our network has not yet been able to learn and all these results are from the dynamics between these networks.

5.3 Generalized winner-takes-all network

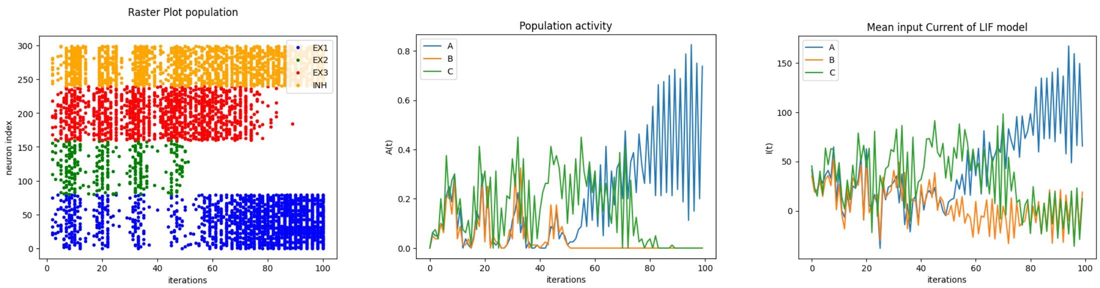
So far, in the context of decision making we only experimented with two decisions, meaning there are only two excitatory populations. Not all decisions are binary so, we can have a generalized network that can make decisions between k different possibilities or excitatory populations. for this purpose we can also use a generalized winner-takes-all dynamics called **effective inhibition**, that simulates the effects and the competition created by the inhibitory population without its implementation.

Effective inhibition (k decisions)

$$\tau \frac{dh_k}{dt} = -h_k + w_0 g(h_k) - \alpha \sum_{j \neq k} g_E(h_j) + RI_k \tag{5.4}$$

Here at first, for simplicity, we try this with only 3 excitatory populations and one inhibitory one to see whether the decision making is still in place. In this experiment, the input current of population (A) is the one to be increased however (C) has an initial higher input current comparing to (B).

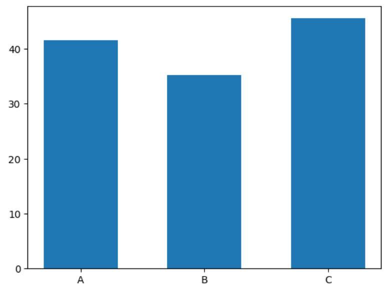
As we can see in Figure 5.4, the decision maing process still works just fine. At first all of them have around the same activity which goes back and forces but as the external input current of (A) increases the weakest link (B) stops spiking however due to the stronger input current of (C) is steal has spikes even after the new period of new input current of (A) for a while, but finally (A) overcomes the other and becomes the final decision of the network as we intend it to do. we can have a better view of this activites



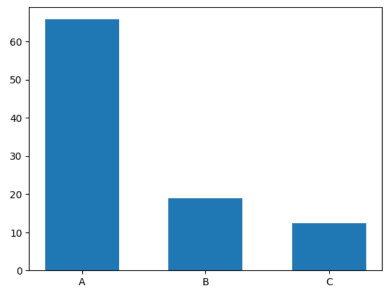
(a) *raster plot of neurons in the network* **(b)** *population activity neurons in the network* **(c)** *mean input current of neurons in the network*

Figure 5.4: *Decision making network of 300 neurons ($k = 3$)*

by comparing the initial and final population activites and accumulative input current.

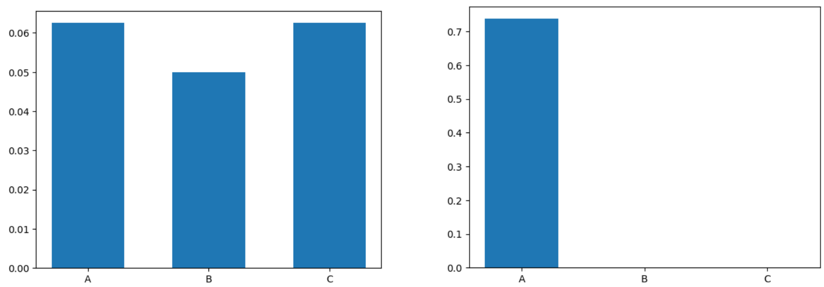


(a) *initial input current of populations*



(b) *final input current of populations*

Figure 5.5: *comparison between initial and final input current of decision making network ($k = 3$)*

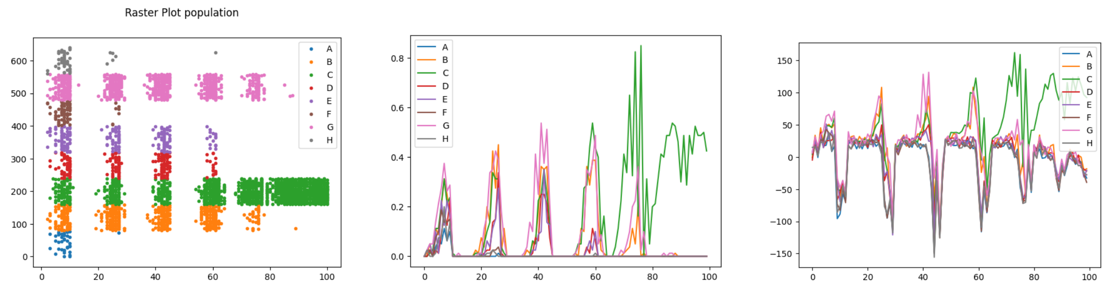


(a) *initial population activity of populations* **(b)** *final population activity of populations*

Figure 5.6: *comparison between initial and final population activity of decision making network ($k = 3$)*

It's completely clear that the decision making works in both changes of population activites and input current. Now let's try in a more general sense, by using bigger value k, setting $k = 8$. Here we add more input current to population (C) to declare it as the winner and we want it to be the final decision.

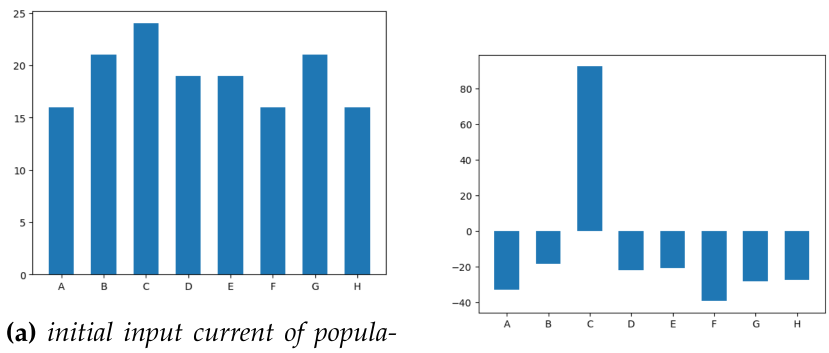
Figure 5.7 shows the results of our experiment. The network even with bigger values of k works just fine. From the raster plot of all populations we can see that based on their external input current they have different spike density but after the increase in



(a) *raster plot of neurons in the network* **(b)** *population activity neurons in the network* **(c)** *mean input current of neurons in the network*

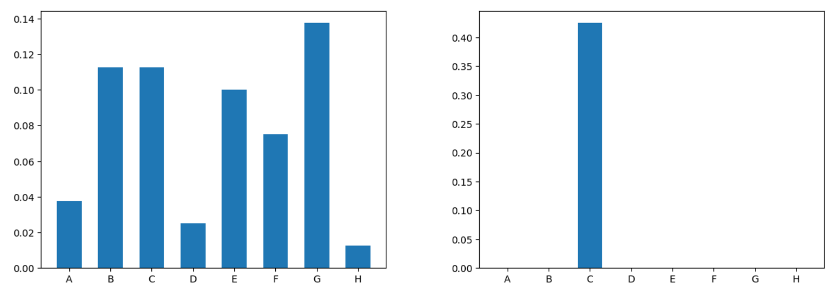
Figure 5.7: *Decision making network of 800 neurons ($k = 8$)*

external input current the behavior of the whole network changes and goes towards higher activity of (C) and lowering the activities of the rest of the populations. as in the previous experiment for better understanding and a clear view of the decision, let's take a look at the initial and final input current and population activities.



(a) *initial input current of populations* **(b)** *final input current of populations*

Figure 5.8: *comparison between initial and final input current of decision making network ($k = 8$)*



(a) *initial population activity of populations* **(b)** *final population activity of populations*

Figure 5.9: *comparison between initial and final population activity of decision making network ($k = 8$)*

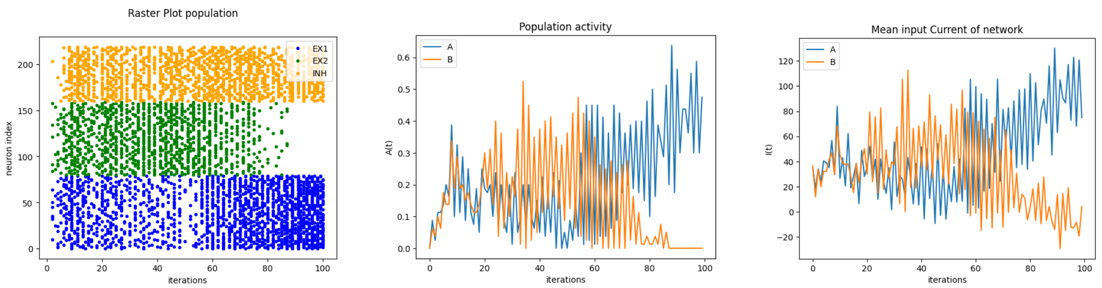
As we can see in Figure 5.9, the same as the previous experiment, the initial accumulative input current is proportionate to the initial external input currents and the population activities are somewhat random however the final states of population activies and input currents clearly shows the winner of the competition and the final decision which would be (C) as we intended.

5.4 In-depth analysis

In the final section of this report, we are going to have an in-depth analysis on the decision-making network we created and try different combinations of parameters and connectivity to bring out different behaviors from this network and analyse them. So far, we have worked with change of final decisions and a generalized version of this network. For simplicity and better understanding of each factor we are going to stick with the basic two-option decision making network for the most part, if necessary, we use the generalized version.

5.4.1 Synaptic weights

One important parameter in a network is its synaptic weights between each of its homogenous populations. We can see the effects of these by lowering the **inhibitory weights** to see the effects on the network.



(a) *raster plot of neurons in the network*

(b) *population activity neurons in the network*

(c) *mean input current of neurons in the network*

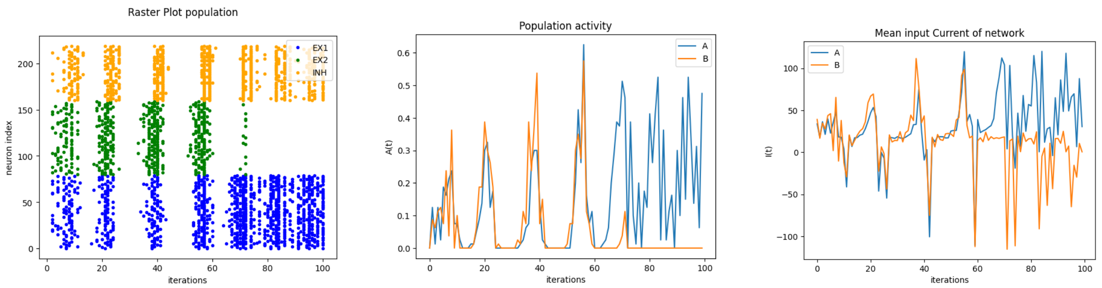
Figure 5.10: *Decision making network of 100 neurons (80/20) - low inhibition*

As we can see in Figure 5.10, the decision making is still working just fine, however simply due to the decrease in inhibition, the activities of the other population (B) took more time to be overcomed resulting in much more activity in the timeline. So what happens if you increase the weights of inhibitory neurons?

well Figure 5.11 tells us that the same as before the decision making process is still working however due to the strong inhibition power of the network the general activity of the whole network decreases considerably.

So far the general network weights were unbiased, meaning that the synaptic weights of both excitatory neurons were identical. Now we want to see the effects of different weights to these population. first we increase the weights to only (A) and then we change it to (B) to see the results and check if the baised weights can alternate the final decision of the network.

Now we can see in Figure 5.12, a considerable difference in behavior of the network as the suppression of (A) starts at the very first iterations and it starts to overcome (B) in

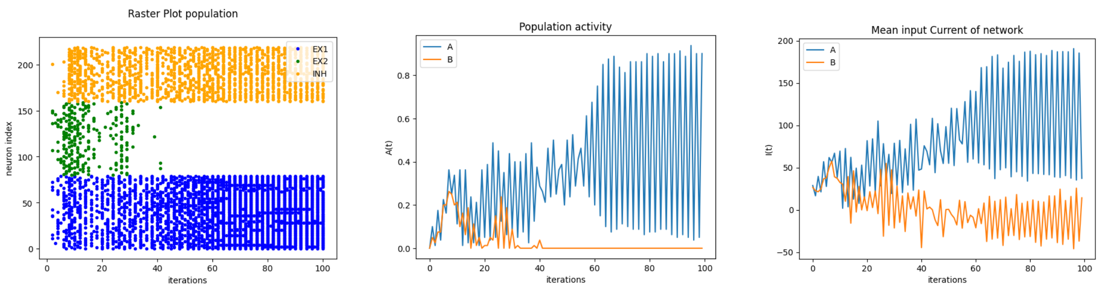


(a) *raster plot of neurons in the network*

(b) *population activity neurons in the network*

(c) *mean input current of neurons in the network*

Figure 5.11: *Decision making network of 100 neurons (80/20) - high inhibition*



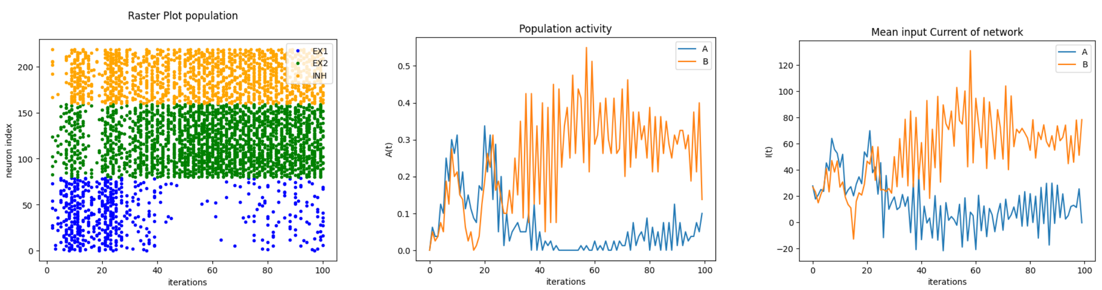
(a) *raster plot of neurons in the network*

(b) *population activity neurons in the network*

(c) *mean input current of neurons in the network*

Figure 5.12: *Decision making network of 100 neurons (80/20) - high weights on (A)*

population activity and of course after the new input pulse, The activities of population (B) is completely wiped out which is a little concerning due to the fact that even without additional input current population (A) was able to somewhat suppress (B) which should not happen however at the end of the day the final decision of the network is still right. Now we increase the synaptic weights of (B) to see the change of behavior.



(a) *raster plot of neurons in the network*

(b) *population activity neurons in the network*

(c) *mean input current of neurons in the network*

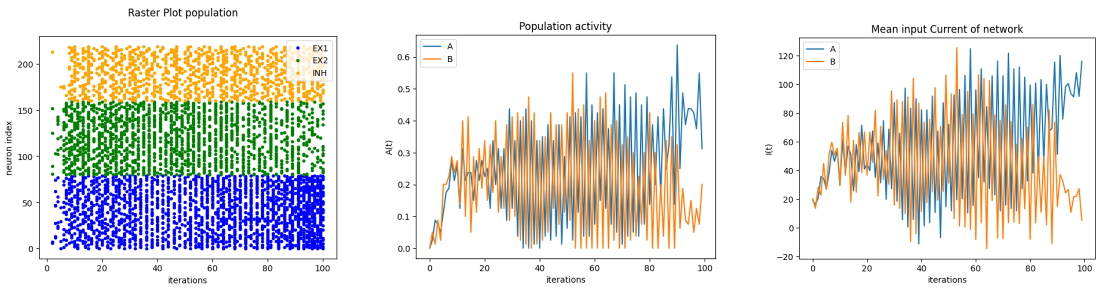
Figure 5.13: *Decision making network of 100 neurons (80/20) - high weights on (B)*

Now, the results of Figure 5.13 is definitely interesting. We intended to declare (A) as the final decision however we can see that the population activity of (B) suppress (A) from the very start making simply due to higher excitatory synaptic weights and we see the effects of additional input current to (A) effects this suppression and (A) starts to have more activity while the activity of (B) starts to decrease but if we want to know the final decision of the network after 100 iterations it would be (B) which is not the desirable result so we can oppose problems in the decision making of the network if the weights are not balanced and equal.

5.4.2 Synaptic connectivity

So far we used the same basic connectivity scheme, as connecting excitatory population to the inhibitory and vice versa with an excitatory synaptic connection between each excitatory population to themselves. Now we can test out the results of the network with different synaptic connectivity.

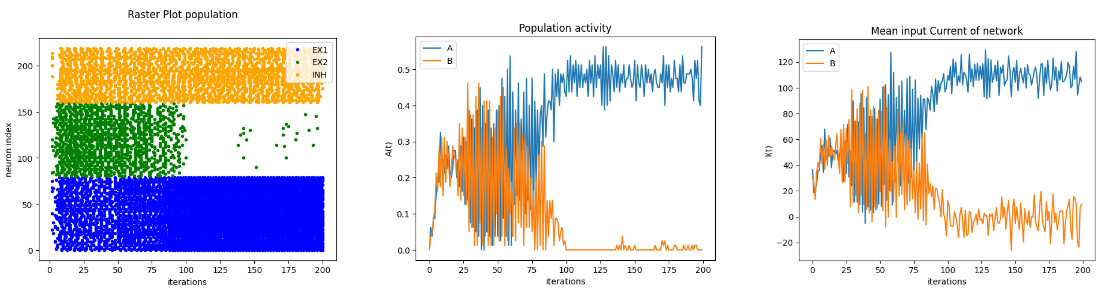
We start with adding an inhibitory synaptic connection to the inhibitory population and observe the results.



(a) *raster plot of neurons in the network* **(b)** *population activity neurons in the network* **(c)** *mean input current of neurons in the network*

Figure 5.14: *Decision making network of 100 neurons (80/20) - (inh-inh connection)*

The plots in Figure 5.14 shows that due to the inhibitory effect of the inhibitory population on itself, the competition between the two networks is less effective, meaning that the suppression of power of population (A) comes after a very long time. It can be seen that both population activity and the accumulative input current of (A) is starting to surpass (B) but the effects are not enough in 100 iterations.

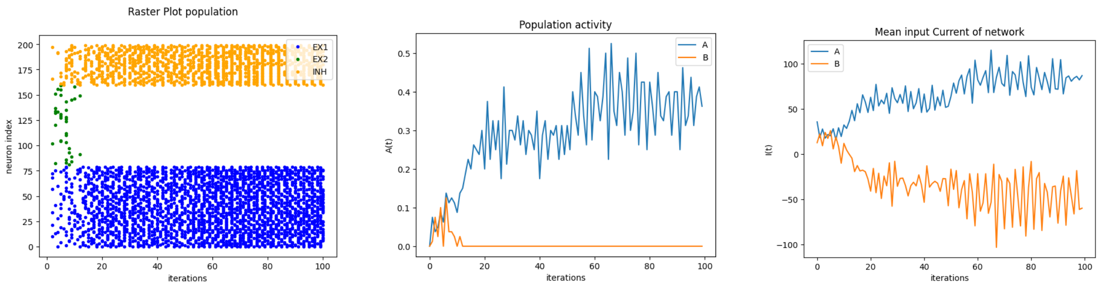


(a) *raster plot of neurons in the network* **(b)** *population activity neurons in the network* **(c)** *mean input current of neurons in the network*

Figure 5.15: *Decision making network of 100 neurons (80/20) - (inh-inh connection) on 200 iterations*

Figure 5.15 is somewhat an extended version of Figure 5.14 with 200 iterations, as we can see this new connection does not effect the final results of the decision but it effects the speed of decision making.

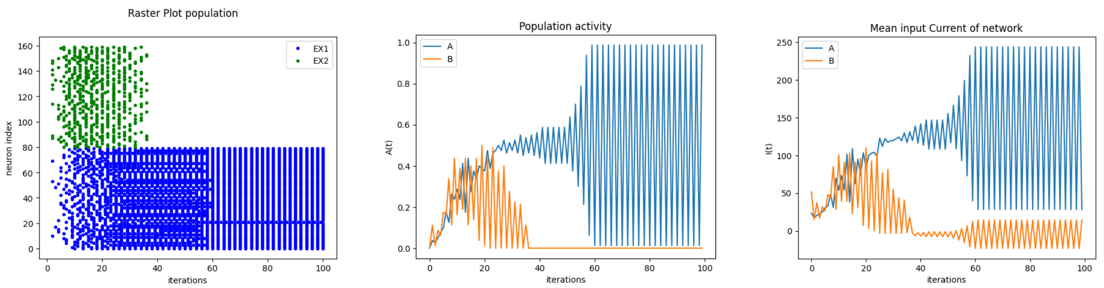
Now this time we are going to add a direct inhibition between two excitatory network beside the inhibition population itself.



(a) *raster plot of neurons in the network* **(b)** *population activity neurons in the network* **(c)** *mean input current of neurons in the network*

Figure 5.16: *Decision making network of 100 neurons (80/20) - (direct inhibition)*

Here we can observe the effects of direct inhibition of the excitatory population on each other. As we can see the final decision has been reached correctly and also fast. So lets see the effects of direct inhibition without the existance of an inhibitory population using the method of **effective inhibition** disccussed in **generalized winner-takes-all networks** on a basic two-option decision making network.



(a) *raster plot of neurons in the network* **(b)** *population activity neurons in the network* **(c)** *mean input current of neurons in the network*

Figure 5.17: *Decision making network of 100 neurons (80/20) - (direct inhibition) without inhibitory population*

Figure 5.17 shows that the method of **effective inhibition** works on a basic decision making network without an inhibitory population. This gives us the abilities to remove the inhibitory population from our calculations and our computation which would make our model faster, on the other hand, for a generalized version we have to keep in mind that due to the removal of the inhibitory population the direct inhibition must be established between each pair of excitatory populations working as different options which needs a computation power on its own. So in order to choose the best option for out decision making networks we should decide with all these factor in mind.

6 Conclusion

In this report, we used our findings and knowledge from the previous project as out building blocks to implement our very first neural populations, we established synaptic connectivity with variety of schemes and compare their behaviors. Next, with help of synaptic connectivity we build a balanced network using an excitatory and an inhibitory neural population, further in this report, we went in-depth to gain a better understanding of each of parameters and components involed in a network. As the final part of this project, we used all of our tools to make a simple decision making network who is set to be one of the basic and yet most important applications of neural networks and had a full analysis of its behavior and its variants. Until now, out populations or neural network do not posses the ability to learn and adapt to the given feedback which is the most fundamental requirement for a real functioning neural network.

This page intentionally left blank.

UNIVERITY OF TEHRAN

NEURAL POPULATION AND DECISION MAKING

ARASH NIKZAD
Student No. 610301195

TEHRAN, MARCH 2024