MASTER THESIS

Entitled

MEMORY AND THE STATISTICS OF CORTICAL ACTIVITY

To obtain

MASTER DE RECHERCHE EN SCIENCES HUMAINES ET SOCIALES

Of the

ÉCOLE DES HAUTES ÉTUDES EN SCIENCES SOCIALES

MENTION: SCIENCES COGNITIVES.

Prepared by
OUSSAMA SABRI

Under the supervision of
GIANLUIGI MONGILLO

September 11, 2017
Declaration

Acknowledgements

I would like to thank my thesis advisor Dr. Gianluigi Mongillo for the knowledge he brings me and the time he devoted to answer my questions. It was a privilege for me to work with him and his superb team during the period of my internship at the "Centre de Neurophysique, Physiologie et Pathologie, UMR 8119" in Paris.

I must express my very profound gratitude to my parents Mina & Mohamed Sabri, my sister Sana and my brother Hossam for providing me with support and continuous encouragement throughout my years of studies. This accomplishment would not have been possible without them. Thank you.

I would also like to thank Pr. Hassan El Amri, my supervisor during the preparation of the master degree on applied mathematics at the ENS of Casablanca from 2014 to 2016, who made me discover the field of computational neuroscience, for his unlimited support and encouragement to reorient to this domain. This modest work is dedicated to you.

Originality

Modeling the memory storage has already been implemented in a recurrent network using a synchronous dynamic where all neutrons’ state is updated at once and the patterns to store were sequences of 0 and 1. As a result, the time evolution of the network dynamics at the stationary state is oscillating between two states or staying in one fixed point. Thus, the network dynamics do not generate temporal fluctuations in the activity of single neurons. The originality of this work is to model the memory storage for which temporal fluctuations of single neurons activity are present and self-generated by the network dynamics itself and the patterns to be stored are probabilities of activation of each neuron in the network.

Contribution

Dr. Gianluigi proposed the present internship. I realized the mean field theory, implemented the learning process for the perceptron and the network. I implemented the simulations and generate the figures and wrote the report. I interpreted the results and discuss them with Gianluigi.
Memory and the statistics of cortical activity.

Background and rationale

Adaptive behavior requires an organism to store information about past events and use it to modify its responses to future ones. According to a major theory, information is encoded and permanently stored by activity-driven changes in the pattern and strengths of the connections between neurons in local networks. The synaptic structure thus generated endows the network with multiple preferred stable states of activity (i.e. attractor states). External inputs (e.g. as a result of stimulation) can cause the network to switch from its current state to any one among its other preferred states. The selection of one such states corresponds to the retrieval of the information passively stored in the synaptic structure.

In the attractor framework, the computational power of the network is directly related to the complexity of its attractor landscape and, in particular, to the number of distinct attractor states that the network dynamics can express (i.e. its storage capacity). This observation suggests that some understanding of the statistical features of the synaptic connectivity in local networks and, presumably, of the resulting neuronal activity, could be gained through optimality arguments. One such optimality arguments – the efficient storage hypothesis – posits that recurrent connectivity, through to-be-identified synaptic plasticity mechanisms, organizes so as to maximize the storage capacity of the network (Brunel et al., 2004; Barbour et al., 2007). Interestingly, the efficient storage hypothesis is able to account for several non-trivial features of the synaptic organization in biological neuronal networks (Brunel et al., 2004; Chapeton et al., 2012; Brunel, 2016)

A paradigmatic example of attractor dynamics in cortical networks is the so-called persistent delay activity (Amit & Mongillo, 2003). Extracellular recordings of neuronal activity in animals performing delayed-response tasks have revealed, in several cortical regions, cells persistently modifying their activity during the delay period. In their basic form, these modifications consist of enhanced/suppressed firing rates as compared to the pre-stimulus period (Fuster & Alexander, 1971; Funahashi et al., 1989). Interestingly, persistent activity is characterized by high levels of spatio-temporal irregularity. Spike trains resemble those generated by Poisson processes (Shinomoto et al., 1999; Compte et al., 2003). The distribution of single-cell average rates is unimodal, right-skewed and long-tailed (Shafi et al., 2007).

The origin and the functional relevance of these high levels of spatio-temporal irregularity for memory functions are presently unclear.

Working hypothesis and key research questions

Our working hypothesis is that high levels of spatio-temporal irregularity in the patterns of neuronal activity are a natural consequence of storing, by means of long-term synaptic modifications, a large number of memories in a distributed neural architecture. In previous studies of the efficient storage hypothesis, network states corresponding to memories are described by binary patterns and the time evolution of the network states is described by a synchronous dynamics, i.e. all neurons’ state is updated at once. As a result, network dynamics in the attractor states cannot generate temporal fluctuations in the activity of single neurons.

In this research, we will extend the efficient storage hypothesis to the case in which the memories to be stored are patterns of single-neuron average activity levels. This clearly requires one to use an asynchronous dynamics (Glauber dynamics). Specifically, we will address the following key questions:

(i) Is it possible to store and retrieve a large numbers of memories (i.e. proportional to the number of neurons in the network) in presence of strong, self-generated spatio-temporal irregularity?

(ii) If so, is the statistics of the resulting synaptic connectivity consistent with experimental observations?
Detailed description of the research project

We will investigate the possibility of storing an extensive number of stable (under an asynchronous update dynamics) fixed points in a network of binary elements by suitably adjusting the connectivity matrix. We will consider a network composed only of inhibitory neurons that receive an external inputs which is uniform and excitatory. This is the simplest setting in which self-generated temporal fluctuations can arise. We will proceed in steps of increasing complexity.

To start with, we will consider a mean-field description (from statistical mechanics) of the network to obtain a reduced description of the population average activities. For infinitely large networks, the network-averaged cross-correlation goes to zero (see, e.g., Renart et al., 2010; Dahmen et al., 2016). In this limit, the temporal fluctuations in the recurrent inputs are described by a Gaussian process whose variance can be computed, once one knows the distribution of activity levels in the memory states, and the mean and variance of the synaptic efficacies (Barri et al., 2016). One can then numerically compute, by using gradient-descent techniques, the synaptic matrix that makes a given set of memories stable fixed points of the MF dynamics (stability is checked a posteriori). Within the MF approximation, we will:

(i) Estimate the storage capacity of the network as a function of the statistics of the activity in the memory states (i.e., as a function of their level of spatio-temporal irregularity);
(ii) Compare the statistics of the synaptic matrices for noisy memory states with those obtained in (Brunel, 2016) for noiseless memory states.

For numerical simulations, \(N\) we will take values in \(\{10^3, 10^4, 10^5, 10^6\}\). We will repeat the simulation for each \(N\) value to ensure that there is no a significant difference between each repetition. In that case, we will take the average values of results over the number of repetitions. To verify that the simulations corresponds to the theory, we will plot the error in function of the size of the network \(N\). The error must decrease as \(N\) becomes bigger.

As the next step, we will verify that the network-averaged cross-correlation does indeed cancel in the limit of large networks. If this is the case, the MF theory developed above becomes exact (in the same limit) and an useful approximation for large, but finite networks. There are two main reasons to proceed at this verification. First, the theoretical results (Renart et al., 2010; Dahmen et al., 2016) holds when there are no correlations in the synaptic matrix. This, clearly, is not the case when the synaptic matrix has been generated by some learning process. Second, cancellation could still occur in presence of such correlations, but it would be effective only for unrealistically large network sizes. We will then numerically compute the cross-correlations matrix (using the method described in (Dahmen et al., 2016)) for networks of increasing size to investigate its limiting behavior.

Finally, if network-averaged cross-correlation becomes negligible in networks of realistic size, we will proceed to numerical simulations of the full microscopic networks. Numerical simulations will be used to verify the (accuracy) of the predictions of the MF theory. Comparison between theoretical predictions and numerical simulations will be validated by using standard statistical analysis techniques.

Expected outcomes

A positive answer to both our research questions will significantly increase the credibility of current, mechanistic theories about the encoding and retrieval of long-term memories. It will also pave the way to a principled understanding of the functional relevance of high levels of spatio-temporal irregularity in patterns of neuronal activity during memory retrieval. A negative answer, on the other hand, would suggest that correlations are key for memory functions. It would thus lead to significant amendments of the current theoretical framework. In either case, our result will advance the understanding of the relationships between synaptic organization and patterns of neuronal activity at the local circuit level, and their theoretical implications for the computational architecture of memory.
Contents

1 Introduction 3

2 Modeling neural network 5
   2.1 The network model 5
   2.2 The network dynamics 6
   2.3 Mean Field Equations 7
      2.3.1 The statistics of the input 7
      2.3.2 Spatial and temporal variability 8
      2.3.3 Balanced state condition 9
      2.3.4 Local average activity 10
   2.4 Numerical simulations 11

3 Self-consistent learning 14
   3.1 Learning a perceptron 14
   3.2 Maximal storage capacity 16
      3.2.1 Numerical simulations 17
   3.3 Learning the Network 20
      3.3.1 Numerical simulation 21

4 Conclusion 30
Chapter 1

Introduction

Memories are thought to be stored in cortical networks by activity-driven changes in the patterns and strengths of the synaptic connections between neurons in the network. Retrieving a stored memory corresponds to evoke the same configuration of the network, i.e. the patterns of activity, that is similar to that occurred when it is stored. One standard way to model memory storage and retrieval mathematically is by making use of attractor states. The synaptic structure generated by storing a new memory endows the network with multiple stable attractor states. Memory recall occurs when the neurons in the network are activated by a stimulus and converge to the attractor state corresponding to that memory.

![Attractor landscape](image)

Figure 1.1: Schematic of attractor landscape in the space of network states. Black dots represent fixed points of network dynamics. Lines indicate boundaries of basins of attraction around each fixed point. Figure from [2].

Attractor neural networks can maintain a persistent activity pattern like those observed in studies of short-term and working memory framework. This persistent activity is characterized by a high spatiotemporal irregularity.

The present work consists in modeling how to store and recall a large number of patterns of activity in a network that self-generate spatio-temporal fluctuation as observed in in vivo. The idea consists in considering that every neuron in the network as a postsynaptic of cells that synapse to it (a perceptron) and at the same time this neuron contributes to the input of another cell, this means that we look at the recurrent network as a set of individual perceptrons. The storage of memories in a recurrent network consists then in adjusting the synaptic distribution for every perceptron to learn to associate the patterns of activity of these memories. Since we
would like to model the storage of memories in a context for which the temporal fluctuations are present and self-generated by the dynamic of the network, we consider an asynchronous dynamic and the patterns to be stored correspond to the probabilities of activation for each neuron so that the neurons’ state is fluctuating with time.

To do so, we proceeded as follow
We considered a network composed of $N$ inhibitory neurons connected at random, with probability $C$, for which each neuron receives an external excitatory input and fixed in time. Since we want the activity to be irregular, we consider an asynchronous dynamic where at each time-step a neuron chosen randomly is updated according to the input it receives, so that at each time cycle the state of the network become stochastic, contrary to synchronous where the dynamics stay in one state or oscillating between two states, in this case the activity is regular. This implies that the patterns of activity to be stored should be probability of activations of neurons so that the spiking becomes irregular.

A summary of the chapters content is given in the following
In the second chapter, we will present the binary neural network model on which the work is build upon. Then we derive a mean-field description of the network in the limit of infinite large networks. Then we give a scaling for which the network is in a balanced state for which neurons exhibit persistent activity with a high degree of variability. We will then perform numerical simulations to verify the predictions of the mean-field theory.

In the third chapter, we will learn a perceptron to store a large number of pattern of activity. The input/output patterns are randomly chosen from the distribution of the activities in the mean field equations with the statistics of the input and its noise are already known. Then we will compute the capacity of the network in the presence of the noise. Then we will train the perceptron to learn a large number of patterns (we will take the capacity inferior to that calculated theoretically because of the finite size network effect). Next we will pass to learn a recurrent network.
Chapter 2

Modeling neural network

The first part of this chapter, we will present the binary-neural network model, on which the work is built upon, then we give the expression of the population average activity of the network that satisfies a first order differential equation in function of the average of the mean input. Then we give the mean field description of the input to characterize its statistics at equilibrium (at the stationary state). After deriving a scaling for which the network is in a balanced state, we proceed to numerical simulations to verify theoretical predictions of the statistics of the input.

2.1 The network model

Neuron model

We consider a binary neuron model by which its state can be expressed by a binary variable, denoted $S$, being either 1 or 0 (active or inactive state respectively). The activation of a neuron depends deterministically on the input it receives, denoted $h$

$$S = \Theta(h - \theta) = \begin{cases} 
1 & \text{if } h > \theta, \\
0 & \text{otherwise.}
\end{cases}$$

Where $\Theta$ is the Heaviside step function (Fig.2.1) and $\theta$ is a fixed threshold.

Network model

We consider a neural network composed of $N$ inhibitory neurons such that each neuron $i$ in the network receives input from its postsynaptic cells (inhibitory input) plus an external excitatory input denoted $h_{\text{ext}}$ fixed in time. In the following, every quantity relative to the $i^{th}$ neuron will be indexed by a subscript $i$. We denote by $w = (w_{ij})_{1 \leq i,j \leq N}$ the synaptic strength matrix, with $w_{ij} \geq 0, \forall i,j \in \{1,\ldots,N\}$. The architecture of the network is determined by the connectivity matrix $(c_{ij})_{1 \leq i,j \leq N}$, for which every neuron $i$ has the probability $C$ to receive a connection from another neuron $j$ in the network, with the convention $c_{ii} = 0$

$$c_{ij} = \begin{cases} 
1 & \text{with probability } C, \\
0 & \text{with probability } 1 - C.
\end{cases}$$

Figure 2.1: The activation/inactivation is determined by the evaluation of the input $h$ with respect to $\theta$. 
$c_{ij} = 1$ means that neuron $j$ projects a connection to neuron $i$, otherwise $c_{ij} = 0$. This connectivity is fixed in time once a particular realization is determined.

The net afferent input $h_i$ received by the $i^{th}$ neuron depends on the states of neurons that are connected with it proportionally to their synaptic weights. It is expressed as follow

$$h_i = - \sum_{j=1}^{N} c_{ij} w_{ij} S_j + h_{ext}. \quad (2.1.1)$$

We next define a dynamic to express the time evolution of the system.

### 2.2 The network dynamics

The network model described above has $2^N$ possible configurations $X = (S_1, \ldots, S_N)$, each of which represents a binary vector whose elements are the states $S_i$ of each neuron $i$. A transition rule is needed to describe the time evolution of the network state from one configuration to another. Therefore, we consider an asynchronous dynamics (Glauber dynamics) for the network update. In what follow, we will consider two time scales defined as:

- A microscopic time scale that we will call time-step, denoted $\delta t = 1/N$, in which one neuron updates its state in that period.

- A macroscopic time scale that we will call time-cycle, denoted $t$, in which the network state is updated. It is equivalent to $N$ individual updates i.e. after $N$ time-steps.

Therefore, the network update using the asynchronous dynamic is done as: in an interval $[t, t+1]$ of two time-cycles, it is composed of $N$ time-steps, at each time-step a neuron is chosen randomly and update its state according to

$$S_i(t + (n + 1)\delta t) = \Theta(h_i(t + n\delta t) - \theta) = \begin{cases} 1 & \text{if } h_i(t + n\delta t) > \theta, \\ 0 & \text{otherwise}. \end{cases} \quad (2.2.1)$$

Where $i$ is the neuron chosen to be updated at the time-step $t + (n + 1)\delta t$, and $n \in \{1, \ldots, N\}$ is the number of updates made since the time $t$. One has: $t + N\delta t = t + 1$.

In what follow, we will omit to write $\theta$ because it can be absorbed by the external input $h_{ext}$.

**Local and population average activity dynamics**

We call the local average activity of a neuron $i$ at time $t > 0$, denoted $m_i(t)$, expressing the probability of activation of the neuron $i$ as $t \to \infty$, the time average of the activity $S_i(t)$. It is expressed as follow

$$m_i(t) = \frac{1}{t} \sum_{n=0}^{t} S_i(t_n). \quad (2.2.2)$$

It has been shown in [8] that $m_i$ satisfies a first order differential equation in function of the input $h_i$ received by $i$. It is given as follow

$$\frac{\partial m_i}{\partial t}(t) = -m_i(t) + \langle \Theta(h_i) \rangle(t). \quad (2.2.3)$$
The expectation value $\langle \cdot \rangle$ can be interpreted as an average over all possible configurations of the network.

At equilibrium in (2.2.3), one has

$$m_i = \langle \Theta(h_i) \rangle. \quad (2.2.4)$$

The fixed point $(m_1, \ldots, m_N)$ can be interpreted as the distribution of the mean activity of individual neurons during the delay period in the delayed-response task.

Knowing the statistic of the input $h$ in the network, one can compute the value of $m_i$ and their distribution. For that, we derive the mean field equations to characterize its statistic.

In what follow, we denote by

$$m = \frac{1}{N} \sum_{i=1}^{N} m_i,$$

the population average activity. It represents the proportion of active neurons in the network.

### 2.3 Mean Field Equations

To characterize the statistics of the total input, we mention that its variability is determined by two components which are the variability of spatial and temporal fluctuations. The first one is due to the identity of neuron and their specific synaptic connections, thus the output varies from one neuron to other. The second one is due to the stochastic flipping of neurons over time, so the output is changing continuously as the neurons flip their states.

#### 2.3.1 The statistics of the input

Let $i \in \{1, \ldots, N\}$. The input received by $i$ at time $t$ is given by

$$h_i = - \sum_{j=1}^{N} c_{ij}w_{ij}S_j + h_{ext}. \quad (2.3.1)$$

$h_i$ is sum of random variables $Y_j = c_{ij}w_{ij}S_j$, $j = 1, \ldots, N$. As we consider sparse a connectivity, then any couple of neuron will have a weak probability to share the same afferent input coming from the same presynaptic neurons, and such correlation is very weak\(^\text{[7]}\) for the large $N$ limit. Thus, one can consider that $Y_j = c_{ij}w_{ij}S_j$ as i.i.d. random variables. Hence the mean and variance of input $h$ is as follow

$$E(Y_j) = \langle c_{ij}w_{ij}S_j \rangle,$$

$$= \langle c_{ij} \rangle \langle w_{ij} \rangle \langle S_j \rangle,$$

$$= C \langle w \rangle m.$$

Considering the fact that $c_{ij}^2 = c_{ij}$ and $S_j^2 = S_j$, one has

$$\text{Var}(Y_j) = E(Y_j^2) - (E(Y_j))^2,$$

$$= \langle c_{ij}^2 w_{ij}^2 S_j^2 \rangle - (C \langle w \rangle m)^2,$$

$$= \langle c_{ij}^2 w_{ij}^2 S_j \rangle - (C \langle w \rangle m)^2,$$

$$= C \langle w^2 \rangle m - (C \langle w \rangle m)^2.$$
Since $h$ is sum of i.i.d random variables with the same mean and variance, $h$ is approximated by a Gaussian distribution via the central limit theorem as $N \to \infty$ with mean and variance as follow
\[
\begin{align*}
\mu &= -NC\langle w \rangle m + h_{\text{ext}}, \\
\sigma^2 &= N(C\langle w^2 \rangle m - (C\langle w \rangle m)^2).
\end{align*}
\] (2.3.2)

Now, we are able to express the population average activity $m$ in function of $\mu$ and $\sigma^2$. One has
\[
m = P(h > 0) = \int_{+\infty}^{+\infty} \frac{1}{\sqrt{2\pi}\sigma^2} \Theta(h) \exp\left[-\frac{1}{2}\left(\frac{h-\mu}{\sigma}\right)^2\right] dh,
\]
\[
= \int_{-\mu}^{+\mu} \frac{1}{\sqrt{2\pi}} e^{-z^2/2} \sqrt{2}\pi d = H\left(\frac{-\mu}{\sigma}\right).
\]

Where, $H$ is the complementary cumulative Gaussian function with zero mean and unit variance defined as follow
\[
H(x) = \frac{1}{2} \text{erfc}\left(\frac{x}{\sqrt{2}}\right). \tag{2.3.3}
\]

### 2.3.2 Spatial and temporal variability

The variability $\sigma_q^2$ of the input is influenced by both spatial and temporal variability denoted as $\sigma_q^2$ and $\sigma_f^2$ respectively. The first is due to the number and the identity of input received by a neuron which explains the spatial variability of input across neurons. The second one is due to the stochastical flipping of postsynaptic neurons, thus the input varies in time.

We introduce the following quantity
\[
q = \frac{1}{N} \sum_{i=1}^{N} m_i^2.
\]

Since $0 \leq m_i \leq 1$, the quantity $q$ is bounded: $m^2 \leq q \leq m$. The equality take place only when all neurons are active ($q = m = 1$) or inactive ($q = m = 0$).

#### Spatial variability

Using the same used in (2.3.1), but this time we are take the time-averaged input
\[
h_i = -\sum_{j=1}^{N} c_{ij} w_{ij} m_j + h_{\text{ext}} = -\sum_{j=1}^{N} Y_j + h_{\text{ext}}.
\]

The second moment of $Y_j$ is
\[
\langle c_{ij}^2 w_{ij}^2 m_j^2 \rangle = \langle c_{ij} w_{ij}^2 m_j^2 \rangle.
\]
\[
\quad = \langle c_{ij} \rangle \langle w_{ij}^2 \rangle \langle m_j^2 \rangle.
\]
\[
\quad = C\langle w^2 \rangle q.
\]

Hence
\[
\text{Var}(Y_j) = C\langle w^2 \rangle q - (C\langle w \rangle m)^2.
\]

Using the central limit theorem argument, we get
\[
\sigma_q^2 = NC\langle (w^2)^2 \rangle q - C\langle w \rangle^2 m^2).
\]
Temporal variability

The temporal fluctuation of the input is due to the flipping in state of neurons. It is expressed as follow

\[ h_i(t) = \langle h_i \rangle + \delta h_i(t). \]
\[ \Rightarrow \delta h_i(t) = h_i(t) - \langle h_i \rangle, \]
\[ = - \sum_{j=1}^{N} c_{ij} w_{ij} (S_j(t) - m_j), \]
\[ = \sum_{j=1}^{N} c_{ij} w_{ij} \delta S_j(t) = \sum_{j=1}^{N} Y_j. \]

Remark that \( \langle Y_j \rangle = 0 \), so its variance is exactly its second moment \( \langle Y_j^2 \rangle \).

One has

\[ \langle \delta S_i^2(t) \rangle = m_i - m_i^2. \]

Hence,

\[ \text{Var}(Y_j) = \langle c_{ij}^2 w_{ij}^2 \delta S_j^2(t) \rangle = \langle c_{ij} w_{ij}^2 (m_j - m_j^2) \rangle, \]
\[ = \langle c_{ij} \rangle \langle w_{ij}^2 \rangle \langle (m_j - m_j^2) \rangle, \]
\[ = C \langle w^2 \rangle (m - q). \]

Using the central limit theorem argument, we get

\[ \sigma_q^2 = NC(\langle w^2 \rangle q - C \langle w \rangle^2 m^2). \]

To summarize, one has: \( h \sim \mathcal{N}(\mu, \sigma^2) \) for sufficiently large networks with

\[
\begin{cases}
\mu = -NC \langle w \rangle m + h_{ext}, \\
\sigma_q^2 = NC(\langle w^2 \rangle q - C \langle w \rangle^2 m^2), \\
\sigma_j^2 = NC \langle w^2 \rangle (m - q), \\
\sigma^2 = \sigma_q^2 + \sigma_j^2.
\end{cases}
\] (2.3.4)

2.3.3 Balanced state condition

As we can see in the expressions of \( \mu, \sigma_q^2 \) and \( \sigma_j^2 \), they all depend on the network size \( N \). Thus we need a scaling to ensure that those variables are well defined as \( N \) goes to infinity. For that, we consider that the synaptic weights \( w_{ij} \) are of order \( O(1/\sqrt{N}) \) and the external input \( h_{ext} \) is of order \( O(\sqrt{N}) \).

\[ \langle w \rangle \sim O(1/\sqrt{N}) \text{ and } \langle w^2 \rangle \sim O(1/N). \]

Therefore, on obtains

\[
\begin{cases}
\mu = \sqrt{N}( -C \langle w \rangle m + h_{ext} ) \sim O(\sqrt{N}), \\
\sigma_q^2 = C(\langle w^2 \rangle q - C \langle w \rangle^2 (m)^2) \sim O(1), \\
\sigma_j^2 = C \langle w^2 \rangle (m - q) \sim O(1).
\end{cases}
\] (2.3.5)
In order to get $\mu$ of order 1, the quantity $(-C\langle w \rangle m + h_{ext})$ should be of order $O(1/\sqrt{N})$. Hence the balance condition is

$$m \xrightarrow{N \to \infty} \frac{h_{ext}}{C\langle w \rangle}.$$  

For any finite size network $N$, the population average activity $m$ should satisfy

$$h_{ext} = C\langle w \rangle m - \frac{\sigma H^{-1}(m)}{\sqrt{N}}.$$  

### 2.3.4 Local average activity

For $i \in \{1, \ldots, N\}$, the mean input $h_i$ can be expressed as

$$h_i(t) = \mu + \xi_i \sigma_q + \eta_i(t) \sigma_f,$$

with $(\xi, \eta(t)) \sim \mathcal{N}(0, 1)$.

Thus, The local average activity in the stationary state can be expressed as

$$m_i = \mathbb{P}(h_i \geq 0) = \mathbb{P}(\mu + \xi_i \sigma_q + \eta_i \sigma_f \geq 0)$$

$$= \mathbb{P}(\eta_i \leq \frac{\mu + \xi_i \sigma_q}{\sigma_f}) = \int_{-\infty}^{\frac{\mu + \xi_i \sigma_q}{\sigma_f}} \frac{e^{-x^2/2}}{\sqrt{2\pi}} dx,$$

$$= \int_{-\infty}^{+\infty} \frac{e^{-x^2/2}}{\sqrt{2\pi}} dx = H\left(-\mu + \xi \sigma_q / \sigma_f\right).$$

One can also express the order parameters of the network $m$ and $q$ in function of $\mu$, $\sigma_q$ and $\sigma_f$ as

$$m(\mu, \sigma_q, \sigma_f) = \int_{\mathbb{R}} \frac{e^{-\xi^2/2}}{\sqrt{2\pi}} H\left(-\mu + \frac{\xi \sigma_q}{\sigma_f}\right) d\xi,$$  \hspace{1cm} (2.3.6)

$$q(\mu, \sigma_q, \sigma_f) = \int_{\mathbb{R}} \frac{e^{-\xi^2/2}}{\sqrt{2\pi}} \left[H\left(-\mu + \frac{\xi \sigma_q}{\sigma_f}\right)\right]^2 d\xi.$$  \hspace{1cm} (2.3.7)

It is clear that $m$ and $q$ correspond respectively to the first and second moment of the probability distribution of activity.

### Local average activity distribution

We know that the network shows a spatial irregularity. Indeed, this induce an inhomogeneity in the mean activity of each neuron (inhomogeneity in the local activities). Thus one can see it as a random variable which the distribution can be calculated. One has from (2.3.6)

$$m(\xi) = H\left(-\mu + \frac{\xi \sigma_q}{\sigma_f}\right) = f(\xi).$$

Let denote $G$ the distribution function of $\xi \sim \mathcal{N}(0, 1)$ (it is a Gaussian function with zero mean and unit variance). Thus, the distribution of the local average activity can be computed using the following expression

$$\rho(m) = G\left(f^{-1}(m)\right) \frac{1}{\left|\frac{df}{d\xi}\right|_{f^{-1}(m)}}.$$
It reads

$$\rho(m) = \frac{1}{\beta} \exp \left[ \frac{1}{2} \left( (\alpha + \beta \xi(m))^2 - \xi(m)^2 \right) \right],$$  \hspace{1cm} (2.3.8)

where

$$\xi(m) = f^{-1}(m) = \frac{\sqrt{2} - \text{erfc}^{-1}(2m)}{\alpha}, \quad \alpha = \frac{-\mu}{\sigma_f}, \quad \text{and} \quad \beta = \frac{\sigma_q}{\sigma_f}.$$ 

We plotted in figure below the distribution $\rho$ for four different values of the population average activity $m$ using the parameters in Sec.2.4, and see how the qualitative differences between them.

![Figure 2.2: Examples of local average activity distribution $\rho(m)$ for different values of the population average activity $m$. Calculated from (2.3.8) with parameters in Sec.2.4.](image)

### 2.4 Numerical simulations

A simulation of the network described above with $N = 2 \times 10^4$ randomly connected neuron with probability $C = 0.1$. The weight matrix was generated using a uniform distribution ($w \sim \mathcal{U}([0,1]))$ with a scaling of order $1/\sqrt{N}$. The population average activity is $m = 0.15$. We computed theoretically the expected statistics of the input $\mu = -0.073$, $\sigma_q = 0.034$ and $\sigma_f = 0.061$, the order parameter $q = 0.037$ and the external input verifying the balanced state condition $h_{ext} = 0.007$ (without scaling).
Figure 2.3: Time evolution of the population average activity (top) and the mean input (bottom) of the network in the stationary state. Both activities were taken in a time interval between 50 and 500. The simulation was initialized with $m(0) = 0.4$ and the expected equilibrium is $m = 0.15$.

Fig.(2.4) shows the activity of a single neuron (Top) chosen randomly and the input it receives (bottom). One can observe that the spiking is irregular. Note that the neuron does not spike even if the input that it receives is above the threshold. It spikes only when its afferent input is above the threshold when it has been selected to update in the time-step sequence.

Figure 2.4: Top: Example of the activity of a neuron chosen randomly in the network. Bars represent the spikes at the moments they occur. Bottom: The time evolution of the input received by the same neuron in the simulation. Both activity was recorded in a time interval between 0 and 1000.
Theoretical predictions for the statistics of input (the mean $\mu$ and the variance $\sigma^2$) are very well matched by the simulation. Fig.2.5 left shows the histogram of the input in the network, with a Gaussian distribution obtained by the mean field theory (blue curve) and the one obtained by simulation (black curve). The theory predicts also the spatial variability of local average activity as we can see in Fig.2.5 right.

Figure 2.5: Distribution of input $h$ and the activity $m$. **Left:** Red bars represents the distribution of the input in the simulation in a time interval between 50 and 500 which approximate a Gaussian distribution with estimated mean $\mu_{\text{sim}}$ and variance $\sigma^2_{\text{sim}}$ (black curve). This distribution approaches the expected one from the MF theory (blue curve). **Right:** Red bars represents the distribution of activity in the simulation. The activity was obtained by taking the local average activity of each neuron in a time interval between 50 and 500. The distribution is well approximated by theoretical one (Blue curve).
Chapter 3

Self-consistent learning

In this chapter, we will train a feed-forward network (a perceptron) to store a large number of fluctuating patterns of activity. The storage is done by adjusting the synaptic distribution between the input and output to associate the data with the desired results, then we will consider a recurrent network to store patterns of activity. The patterns to store in the recurrent network are composed of a distribution of activity. The storage in this case is done by considering each neuron as an output of a perceptron and then perform the learning process for all the neuron. Once the learning is achieved, one verify if the network can produces the same activity stored on it.

3.1 Learning a perceptron

Let’s consider a $N$—input perceptron with a single target neuron as illustrated in Fig.3.1. We want the perceptron to learn to associate $p \in \mathbb{N}^*$ fluctuating patterns in input to the desired $p$ fluctuating outputs. Each input pattern, $1 \leq \nu \leq p$, is composed by a $N$—vector of probabilities of activation of each neuron ($m_1^\nu, \ldots, m_N^\nu$) and the output is also composed of $p$—vector of probability for each stored pattern ($m_1^0, \ldots, m_p^0$). Hence, the learning task is to adjust the synaptic weights $w = (w_1, \ldots, w_N)$ so that when the input is fluctuating with probability $(m_i^\nu)_{1 \leq i \leq N}$, the output is fluctuating with probability $m_0^\nu$.

![Figure 3.1: Scheme of the perceptron. The probability of activation of the target neuron is determined by the input it receives evaluated by the activation function $H$.](image)

Since we want to store patterns of activity in a recurrent network, in this case each neuron is considered as an output of its presynaptic cells and the same neuron itself contributes in the
input of its postsynaptic cells, we consider the patterns of the input and output of the perceptron to have the same statistics. In what follow, we denote the local average activity of the $N$ neuron in the input layer in pattern $\nu$, $m^\nu_j$, with a subscript $j$ and a superscript $\nu$. The same notation for the target neuron, but with subscript $0$.

Let’s consider the following set of $p$ input/output

$$\left\{ (m^\nu_j)_{1 \leq j \leq N}, m^\nu_0 \right\}_{1 \leq \nu \leq p}.$$  

These patterns are obtained from the distribution of the local average activity $\rho$ in (2.3.8) such that its statistics is already known.

The input received by the target neuron in pattern $\nu$, denoted $\hat{m}^\nu_0$, is given by

$$h^\nu_0 = -\frac{1}{\sqrt{N}} \sum_{j=1}^{N} w_j m^\nu_j + \sqrt{Nh_{\text{ext}}} \quad (3.1.1)$$

where $w_j \geq 0$ and $h_{\text{ext}} > 0$ are both of order $O(1)$.

The temporal variation is given by

$$\sigma_f^2 = \langle w^2 \rangle (m - q) \quad (3.1.2)$$

Given a vector of synaptic weights $w = (w_j)_{1 \leq j \leq N}$, the output of the perceptron in pattern $\nu$ is as follow

$$\hat{m}^\nu_0 = P(h^\nu_0 > 0) = \int_{-\frac{h^\nu_0}{\sigma_f}}^{+\infty} \frac{e^{-z^2/2}}{\sqrt{2\pi}} \, dz = H\left(-\frac{h^\nu_0}{\sigma_f}\right). \quad (3.1.3)$$

Hence, the probability of activation of the target neuron is determined by the input it receives and on the temporal fluctuation (See Fig. 3.2). In the absence of noise (when $\sigma_f \to 0$), the activation function $H$ approaches Heaviside step function, and the activation becomes deterministic: the neuron is active if $h^\nu_0 > 0$, silent otherwise.

![Figure 3.2: The activation function $H$ for different value of the temporal fluctuation $\sigma_f$. Not that the value of $\sigma_f$ determines the slope of of the curve: the slope increases (resp. decreases) as $\sigma_f \to 0$ (resp. $\sigma_f \to \infty$).](image_url)
The obtained outputs \((\hat{m}_0^\nu)_{\nu=1}^p\) are, in general, different from the desired ones \((m_0^\nu)_{\nu=1}^p\). The learning task of the perceptron is thus to adjust the synaptic weights \((w_j)_{1\leq j \leq N}\) such that each output approaches the desired one for all the patterns. This is equivalent to minimize the following error function

\[
E(w) = \frac{1}{2} \sum_{\nu=1}^{p} (\hat{m}_0^\nu - m_0^\nu)^2 = \frac{1}{2} \sum_{\nu=1}^{p} \left( H\left(-\frac{h_0^\nu}{\sigma_f}\right) - m_0^\nu\right)^2. \tag{3.1.4}
\]

Here the statistics of the output is determined by \(h_0^\nu\) (3.1.1) and \(\sigma_f^2\) (3.1.2) that we do not know because they depend on the synaptic distribution \((w_j)_{1\leq j \leq N}\). However, since we chose the input and output data from a distribution of local average activity that is generated with a specific temporal fluctuation \(\sigma_f^2\), therefore the value of the quantity \(\langle w^2 \rangle\) is known. Hence, we impose a learning constraints in the learning process on the synaptic weights to ensure that the obtained synaptic distribution in the will generate the same temporal variability we have chosen at the beginning to generate the data. We add to (3.1.4) the following constraints

\[
\frac{1}{N} \sum_{j=1}^{N} w_j^2 = \langle w^2 \rangle \quad \text{and} \quad w_j \geq 0, \quad \forall 1 \leq j \leq N. \tag{3.1.5}
\]

The perceptron has to adjust its weights \(w_j\) so that when the input is fluctuating with probability \(m_f^\nu\) its output is fluctuating with probability \(m_0^\nu\). We expect that during the simulation of the perceptron initialized with a binary variable corresponding to the probability of activation of a specific pattern of activity \((m_1^\nu, \ldots, m_N^\nu)\), the average activity of the target neuron \(\hat{m}_0^\nu\) will approach \(m_0^\nu\) with temporal variability \(\sigma_f\) already imposed in learning.

### 3.2 Maximal storage capacity

One may wonder, once the network parameters are given, how many patterns can we store (the capacity of the network), i.e. what is the maximal value of \(p\) such that there exists a solution for the minimization problem (3.1.4)-(3.1.5). To answer to this question, we define the capacity, denoted \(\alpha\), as the ratio of learned patterns \(p\) and the size of the network \(N\)

\[
\alpha = p/N.
\]

So the storage capacity is defined by a critical number of patterns \(p_c\) such that beyond this number, there is no longer any solution for the synaptic weights to store that number of patterns. We denote this value as \(\alpha_c = p_c/N\). This capacity can be calculated analytically for infinitely large networks.

We are interested in the calculation of \(\alpha_c\) with an added learning constraint of imposing \(\langle w^2 \rangle\) because it determines the temporal variation \(\sigma_f^2\). The capacity reads (G. Mongillo, unpublished results)

\[
\alpha_c = \frac{A(B)}{\langle w^2 \rangle + \sigma_q^2 + \sigma_m^2}, \tag{3.2.1}
\]

where \(\sigma_q^2\) and \(\sigma_m^2\) are respectively the variances of the total input and of the activity defined as

\[
\sigma_m^2 = \int_0^1 (m - z)^2 \rho(z) dz,
\]

\[\sigma_q^2 = A(B)\]
where $\rho$ is the distribution of activities (2.3.8).

The function $A$ is defined as

$$A(x) = \left( \frac{\langle w \rangle H(x)}{G(x) - xH(x)} \right)^2,$$

where $H$ as defined in (3.2) and $G$ is the Gaussian function with zero mean and unit variance. The value of $B$ in (3.2.1) is the solution of the following equation

$$\frac{H(B)(1 + B^2) - BG(B)}{(G(B) - BH(B))^2} - \frac{\langle w^2 \rangle}{\langle w \rangle^2} = 0.$$

Once we have the statistics of the network, we can plot the capacity versus $\langle w^2 \rangle$. In fig.3.3, we plot the capacity of the network as a function of $\langle w^2 \rangle$ using the statistics obtained via the mean field theory with fixed parameters $m = 0.15$ and $\langle w \rangle = 0.75$. For every value of $\langle w^2 \rangle$, there exists one value of $\alpha_c$ for which there exists a unique solution satisfying the learning constraints. In what follow, we will chose $\langle w^2 \rangle$ for which $\alpha_c$ is maximum. We note that the synaptic weights to learn the specific set of patterns is unique at capacity in the infinitely large network. In our simulations, the size of the network we are considering is relatively small, and because of the finite-size effects, we will take the capacity below the theoretical capacity.

### 3.2.1 Numerical simulations

We solved the minimization problem for a perceptron with $N = 1500$ neurons and population average activity $m = 0.15$ and the mean of synaptic distribution $\langle w \rangle = 0.75$. We plot the critical capacity for these parameters, and we chose $\langle w^2 \rangle \approx 1.75$ such that the capacity is optimal, i.e. the peak value of $\alpha_c$ in Fig.3.3. Once we have these parameters, the statistical parameters of the input such as the mean input $\mu$, $\sigma_q^2$ and $\sigma_f^2$ are computed by resolving the mean field equations in Section 2.3, they are respectively: -0.52, 0.21 and 0.45.

We then solved the minimization problem (the error function evaluated at the end of the process is $E = 2 \times 10^{-15}$), and take a look to the obtained synaptic distribution (see Fig.3.4Left). We found that this distribution is identical to the distribution of synaptic weights in Brunet et al.\cite{3}. It is composed large number of silent synapses (a delta function at zero weights) and a truncated Gaussian at positive weights. The fraction of zero weights represents broken connection because the constraints imposed during learning and the fact that the weights cannot be negative.
Figure 3.4: Left: Synaptic weight distribution for \( N = 1500 \) neurons and \( \langle w^2 \rangle \simeq 1.75 \). The number of patterns is 180 (\( \alpha = 0.12 \)) against theoretical capacity of about 220 patterns (\( \alpha_c \simeq 0.147 \) in Fig.3.3). Inset graph: zoom of the distribution. Here, the synaptic distribution is plotted without scaling and are of order 1. Right: The error \( E(w) \) of the activities resulting from the time evolution of the perceptron and the desired activity versus the total time of the simulation.

We plotted the obtained results \( (\hat{m}_0^\nu)_{1 \leq \nu \leq p} \) from the input data using the resulting synaptic distribution, in the formula (3.1.3), versus the desired ones \( (m_0^\nu)_{1 \leq \nu \leq p} \) in Fig.3.5Left, we found that the resulting patterns of activity \( \hat{m}_0^\nu \) match the desired activity. We next recorded the activity of the target neuron for each pattern of activity by simulating the perceptron by initializing the dynamic with a binary vector corresponding to the probability of activation of each neuron in the input layer for pattern \( \nu \). The update rule is asynchronous, that is, between two time cycles, there are \( (N+1) \) time steps. At each time-step, a neuron was chosen randomly to be updated between the \( N \) neuron composing the input and the target neuron. If the chosen neuron belongs to the input layer, it is updated with respect to its probability of activation, and if the target neuron was chosen, it is updated in function of the input it receives at that time-step (active if \( h_0^\nu > 0 \) and inactive otherwise).

In a total time of \( t = 1000 \), we took the time average of the target neuron’s activity for each pattern, and we plotted them versus the desired output \( (m_0^\nu)_{1 \leq \nu \leq p} \). (see Fig.3.5Right) and we found that they are much correlated (Pearson correlation \( r = 0.99 \)), and the error between obtained results from the simulation and desired ones was \( E = 2 \times 10^{-2} \). This show that if the simulation was initialized with a pattern of activity \( \nu \), the target neuron activity has the desired average activity \( m_0^\nu \). The error \( E \) obtain in this simulation depends on the total time of simulation, because by definition, the mean activity of a neuron \( i \) is given by taking the limit \( T \longrightarrow \infty \) in (2.2.2). Hence, as we increase the duration of simulation, the target neuron average activity approaches its probability of activation, thus the error \( E \) will decrease. So, we plotted the evaluation of the error \( E \) versus the time duration of the simulation (see Fig.3.4Right). Indeed, as expected, we have found the the error \( E \) decreases as the duration of simulation increases.
Figure 3.5: Plot of resulting patterns versus stored ones. Left: $(\hat{m}_0^\mu)_{1 \leq \mu \leq p}$ were obtained using the formula (3.1.3). Here the error is $E(w) = 2 \cdot 10^{-15}$. Right: $(\hat{m}_0^\mu)_{1 \leq \mu \leq p}$ were obtained by taking the target neuron’s mean activity over the duration of the simulation which last 1000 time cycles. Here the error is $E(w) = 2 \cdot 10^{-2}$.

We verify that the target neuron exhibit the same temporal fluctuation $\sigma_f$ as the data. For that, we computed the temporal variability of the input received by the target neuron, that we denote $\sigma_f^\nu$, for each pattern of activity $\nu$ in a time duration of 250 time cycles, and we plotted $\sigma_f^\nu$ versus the theoretical $\sigma_f$ in figure below where we plotted $f(\nu) = \sigma_f^\nu \ast \nu + (\sigma_f - \sigma_f^\nu)$, for $\nu = 1, \ldots, N$ are the identity of stored patterns.

Figure 3.6: Plot of the resulting variability $\sigma_f^\nu$ of input from the time evolution of the perceptron versus theoretical value $\sigma_f$. The duration of simulations permitted to estimate $\sigma_f^\nu$ is 250 time cycle, each one was initialized with pattern $p$. The x-axis is identity of the pattern of activity $\nu$, the y-axis is the evaluation of $f(\sigma_f^\nu)$. Red dots represent the pairs $(\sigma_f^\nu = f(\sigma_f^\nu))$ and the blue curve is linear with the slope $\sigma_f$. 

19
We found that all the obtained $\sigma^\nu_f$ match well the theoretical variability $\sigma_f$. One has:

$$\frac{1}{N} \sum_{\nu=1}^{p} \sigma^\nu_f = 0.449 \simeq \sigma_f = 0.451 \quad \text{and} \quad \frac{1}{N} \sum_{\nu=1}^{p} (\sigma_f - \sigma^\nu_f)^2 \simeq 9 \times 10^{-4}.$$ 

The duration of simulations for which $\sigma^\nu_f$ is estimated, and the finite size network, are behind these little variations between $\sigma^\nu_f$ and $\sigma_f$. One think that these variations goes to 0 as the duration of simulation and the network size go to infinity.

Therefore, one conclude that when the perceptron is fluctuating in pattern $\nu$, the output is fluctuating in the desired pattern $m^\nu_0$ with temporal noise $\sigma_f$.

We plotted examples of activity of the target neuron for four patterns of activity chosen randomly in total time simulation of 500 time cycles in which one can see that the activity is irregular.

Figure 3.7: Example of four activity patterns chosen randomly over 180 stored patterns. Bars represents when the neuron is active ($S^\nu_0(t) = 1$) occurred at their corresponding timing. The parameters are the same as the past figures in this chapter. The mean activity of the target neuron for each pattern is: $m^{33}_0 = 0.258$; $m^{80}_0 = 0.068$; $m^{95}_0 = 0.075$; $m^{146}_0 = 0.187$.

### 3.3 Learning the Network

Let’s consider the network described in Chap. 2. at Each neuron $i$ in the network receives exactly $N' = NC$ input from other neurons. At the same time, $i$ contributes in the input to another neurons. Thus one can see the network as composition of $N$ perceptrons, with $N'$ input layer and a single target for each one.

We are going to train to whole network to store a large number of independent patterns of activity. These patterns are distribution of local average activity $(m^\nu_1,\ldots,m^\nu_N)_{1 \leq \nu \leq p}$. This is done by learning each perceptron by considering the input data as the set of patterns of the postsynaptic cell of the target neuron and the output data are the target neuron’s patterns of activity. If all the perceptrons learn to associate patterns of activity as done in the past sections, we expect that the network will be capable to generate the activity patterns that we stored on it. Each pattern $(m^\nu_1,\ldots,m^\nu_N)_{1 \leq \nu \leq p}$ is a fixed point of the network, since it corresponds to a solution
of the dynamical system (2.2.3), and it corresponds to the attractor state of the pattern \( \nu \) in the phase space of the network.

Once the network parameters \( N, C, m, \langle w \rangle \) and \( \langle w^2 \rangle \) are given, one can derive the statistics of the input by solving the mean field equations under the balance state hypothesis. The fixed points are chosen randomly in the distribution of the local average activity \( \rho(m) \) with statistics determined by the given parameters. So each perceptron has to learn to associate the patterns of activity of the target neuron with the patterns of the \( N' \) neurons that synapse to it. As done in the past section, we will impose a learning constraint on the synaptic weights in order to obtain the same temporal fluctuation involved in generating the distribution of activities \( \rho(m) \).

Let’s consider \( p \) independents patterns of activity. Each pattern is composed of \( N \) probability of activation \( m^\nu_j \) with a temporal variability \( \sigma_f = \sqrt{NC\langle w^2 \rangle(m - q)} \). For each neuron \( i \) in the network receives an input in pattern \( \nu \) as

\[
h^\nu_i = -\frac{1}{\sqrt{N}} \sum_{j=1}^{N} c_{ij} w_{ij} m^\nu_j.
\]

The output the probability of activation of the neuron \( i \) in function of the synaptic weight distribution is thus

\[
\hat{m}^\nu_i = H\left( \frac{-h^\nu_i}{\sigma_f} \right). \tag{3.3.1}
\]

The learning task is thus to adjust the synaptic weights \( (w_{ij})_{1 \leq j \leq N} \) such that \( \hat{m}^\nu_i \) approach the desired one \( m^\nu_i \). This is done by minimizing the cost function

\[
E_i(w) = \frac{1}{2} \sum_{\nu=1}^{p} (\hat{m}^\nu_i - m^\nu_i)^2. \tag{3.3.2}
\]

with a learning constraints

\[
\frac{1}{N} \sum_{j=1}^{N} c_{ij} w_{ij}^2 = C\langle w^2 \rangle \quad \text{and} \quad w_{ij} \geq 0 \quad \forall j \in \{1, \ldots, N\}.
\]

In our simulations, we learned only the synapses that are connected to the target neuron in each perceptron.

### 3.3.1 Numerical simulation

We trained a network of size \( N = 3000 \) randomly connected with probability \( C = 0.1 \) to store \( p = 21 \) (instead of theoretical capacity \( p_c \approx 44 \)) independents fixed points with the same statistics obtained from the distribution of activity in the mean field theory in Chap.2 satisfying the balanced condition with the parameters \( m = 0.15, \langle w^2 \rangle \approx 1.75 \) and \( \sigma_f \approx 0.14 \).

After learning the learning, we looked at the synaptic weights distribution. As expected, it has the shape of truncated Gaussian in the positive weights with large number of silent synapses (Fig.3.8) left. We also plotted the expected patterns of activity resulting from learning by plotting all the fixed points versus those evaluated by \( H \) in (3.3.1) with the obtained synaptic distribution (Fig.3.8) right, and we found that all the obtained patterns \( \hat{m}^\nu_j \) approaches well the stored ones.
Figure 3.8: Left: Synaptic weight distribution obtained after learning 21 patterns. Here we plot only the distribution of connected neurons in the network. Right: Plot the expected pattern of activity $m^\nu_j$ versus obtained ones $\hat{m}^\nu_j$ for all neurons in all patterns using (3.3.1). The mean error obtained in this simulation is $E = \frac{1}{N} \sum_{i=1}^{N} E_i = 7.4 \times 10^{-7}$.

We verified the linear stability of the fixed points of the dynamical system (2.2.3). We recall that a fixed point is stable if all the eigenvalues of the linearized system (the Jacobian matrix) evaluated at the fixed point have all strictly negative real part. We plotted in Fig.3.9 all the eigenvalues of all the fixed points stored in the network in the complex plane, and as one can see, all the eigenvalues have a strictly negative part. This means that all the fixed points stored in the network are stable. For sufficiently large networks, one may only verify the stability of one single fixed point, because all the fixed points chosen here are statistically equivalent. Thus the stability of one fixed point implies the stability of all the other fixed points and vice versa.
Figure 3.9: Plot in all eigenvalues of the stored fixed points in the complex plane. The highest real part value over all the eigenvalues is about $-0.05$.

Once we have the stability of the fixed points, we simulate the network by initializing it by a binary vector with probability chosen randomly. We expect that the dynamics will converge after a transient time to one of the stored fixed points in the network, and thus the activity of the network at equilibrium will be highly correlated to this fixed point.

We plotted in Fig. 3.10 the correlation of the activity at equilibrium of the network initialized with a binary vector with a specific probability $P$. We considered 20 values for these probabilities and we initialize the network for each of these values. We found that at each simulation (Fig. 3.10), the activity correlates with a unique fixed point stored in the network. This means that the dynamics converges to the attractor state corresponding to that fixed point. We then simulated the network 20 times for a same probability of activation of the binary vector used to initialize the network (Fig. 3.10). We found that the activity converges to one of the fixed point, but the identity of the fixed point changes from one realization to another.

The observation we see in both figures when the activity slightly correlates with two fixed points may be explained by the fact, that in that particular realization, the activity switch from one pattern to the other.
Figure 3.10: The correlation of the network activity at equilibrium with all the stored fixed points. Total time is 500 time cycle and the network activity was taken by averaging the activity in a time interval between \( t = 50 \) and \( t = 500 \). Left: lines correspond to the probability of activation of the vector used to initialize the network. Columns are the identity of fixed points. Right: As in the left. Here we repeated 20 times the simulation using the same probability of activation \( P = 0.76 \).

At this stage, we know that the activity of the network converges to one of the stored fixed points. We are wondering if one can recall a specific pattern of activity i.e. if the network can exhibit a specific patterns of activity. One idea is to initialize the network by a binary vector corresponding to the probability of activation of each neuron in a particular fixed point. We expect that the time evolution of the network stay at the same attractor state. To verify this hypothesis, we subdivide the total time of simulation into small time intervals, and compute the correlation of the distribution of the local average activity (averaged on each time interval) with all the fixed points stored in the network. If what we assumed is true, we expect that the distribution of activity correlates with a unique and the same fixed point over all the time intervals. Also we expect that the fixed point which the activity correlates with, is the same fixed point used to initialize the simulation.

We initialized the network with a binary vector with the probability of a fixed point chosen randomly (here, it is the fixed point 12). We chose a time interval of 15 time cycles and run the simulation for a duration of 6000 time cycles. We plotted the correlation of the activity exhibited by the network with all the stored neuron in Fig.3.11, and we found that this activity is largely correlated with a unique fixed point the long time intervals. The fixed point which it correlates is the same fixed point that used to initialize the simulation. The reason we take small interval was the fact that the dynamics of the network may make transition from one attractor state to another because of the noise in finite size networks. Once we know that the network remains in the same attractor state, we calculated the correlation of the activity averaged over the total time cycle with the 12th fixed point and we obtained \( r = 0.91 \).
Figure 3.11: The correlations of the network activity with all the stored fixed points. The curve with black dots corresponds to the fixed point for which the network has been initialized. Here, it is the 12th pattern. The x-axis corresponds to a time interval of 15 time cycles for which the activity is averaged. The total duration of the simulation is 6000 time cycles.

To test that the network remains at the same attractor state that used to initialize it, we initialized the network for each fixed point and compute the correlation of the resulting activity in each pattern with all other stored fixed points. This simulation is repeated 10 times. We calculated the mean correlation $r$ over the 10 realizations by taking the Fisher’s $z$-transformation for each realization given by $z_i = \text{arctanh}(r_i)$ where $r_i$ is the Pearson correlation for a single realization, then we take the average $z$ of $z_i$ over the 10 realizations. The mean correlation $r$ is obtained using the inverse of Fisher’s $z$-transformation as follow: $r = \text{tanh}(z)$. The total time of each simulation is 1000 time cycles and the activity of neurons were averaged over the time duration of the simulation. We found that almost all the activities of the network initialized by a specific pattern of activity remain in the same pattern of activity used to initialize the network over the 10 realizations (see Fig. 3.12 Right). There was a particular patterns (pattern 4 and 15) that exhibit a relatively large correlation with two fixed points. This could be explained by the fact that in some realization of simulations, the activity makes transition from the pattern that is initialized with to another attractor state. This hypothesis is supported by the fact that; as one can see in Fig. 3.12 Left, showing the correlation of the activity for one realization of the simulation above; the network activity initialized with pattern 15 strongly correlates with the fixed point 12 not with the fixed point 15 which it is supposed to correlate with. This means that one moment in the simulation the activity switch to the activity in pattern 12. One has also the activity for the pattern 4 correlates the expected fixed points in the single simulation realization, but it shows correlation with pattern 20 the 10 simulation realizations, thus there was particular realization between the ten realizations for which the activity exhibits correlation for this pattern, thus the network makes transition from pattern 4 to pattern 20.
Figure 3.12: The correlation of the network activity with the stored fixed points. Each line corresponds to the identity of the fixed point for which the network was initialized in simulation and the correlation was evaluated with all the fixed points in columns. *Left:* The correlations are obtained in one realization of simulating the network for each pattern. *Right:* The correlations are obtained using Fisher-transform over 10 realization for each pattern. All the simulations were performed in total time of 1000 time cycles.

Now, we have increased the number of fixed points to store in the network to $p = 30$, using the same parameters above. After learning, we looked at the distribution of synaptic weights, it has the same shape as in Fig.3.8, with an increasing of silent synapses. This is expected because we increased the number of patterns to learn in the network. When we plotted the stored patterns $m^j_\nu$ versus the obtained ones $\hat{m}^j_\nu$, we remarked that there is a fraction of $\hat{m}^j_\nu$, that does not exactly match the desired ones, but the error is still small. Here the mean error is $E \simeq 1.8 \times 10^{-5}$.

Figure 3.13: *Left:* Synaptic weight distribution obtained after learning 30 patterns. *Right:* Plot the expected pattern of activity $m^j_\nu$ versus obtained ones $\hat{m}^j_\nu$ for all $j$ and $\nu$. The mean error is $E \simeq 1.8 \times 10^{-5}$.
After verifying that all the fixed points are linearly stable; the maximum real part of all eigenvalues of all the fixed points was about $-0.025$; we simulated the network to verify if its activity correlates with the fixed point used to initialize it, and if the activity remains in the same attractor state. We proceed as in Fig.3.11, we subdivide the total time into small time intervals, and compute the correlation of the network activity averaged on each time interval with all the stored patterns (see Fig.3.14). Contrary to Fig.3.11, this time we found that the activity of the network starts to exhibit the activity of the pattern for which the network is initialized with, then the activity correlates with many other fixed points at different moments. This suggests that the activity of the network makes transitions from one pattern to another. We calculated the correlation of the activity in the time cycle interval $[1950, 3000]$ (about $[130, 200]$ in Fig.3.14) corresponding to period where the activity remains longer and it was about 0.92, and the fixed point corresponding to that pattern was the $28^{th}$.

![Figure 3.14: The correlation of the network activity with all the stored fixed points. The simulation was initialized with the 9$^{th}$ fixed point represented by the red curve with dots. The time duration of the simulation is 6000 time cycles and the size of each time interval is 15 time cycles.](image)

The transition in pattern observed above can be seen in the activity of a single neuron. As one can see in Fig.3.15, the activity of the neuron is highly activated, thus the mean input it receives is higher, in some periods of time and lower activity, lower mean input in other ones, which means that that neuron has a higher probability of activation in one fixed point and when the activity of the neuron become lower this means that the network makes transition to another fixed point where the mean activity of this neuron is low.
Figure 3.15: Top: The activity of a single neuron. Bottom: The input received by the neuron. The red curve determines the threshold. The time duration is 6000 time cycles.

We increased further the number of patterns to store using the same parameters above to \( p = 36 \) fixed points (the highest eigenvalue real part over all the fixed points is \(-0.06\)), and we plotted the correlation of the network activity. We found the same behavior seen for the case of \( p = 30 \), Fig.3.16 suggests that the network dynamic makes transitions in several stored fixed points. However, as one can see, the dynamics is very unstable, \( i.e. \) the dynamic makes fast transitions and it does not stay at a fixed point longer.
Figure 3.16: The correlation of the network activity with all the stored fixed points. The simulation was initialized with the pattern 29 for a duration of 6000 time cycles. The size of each time interval is 10 time cycles.
Chapter 4

Conclusion

In this work we were able to set learning in a balanced network for which the temporal fluctuations are present and self-generated. The fixed points we stored are stable, and we found that the lifetime of the network activity in an attractor state decreases as to number of stored fixed points increases. The retrieval of a fixed point is done when the network is initialized with a binary vector corresponding to the probability of activation of each neuron in that fixed point. The network dynamics stay at the same attractor state corresponding to that fixed point. As we increase the number of fixed points to be stored, we remark transitions between several state, and these transitions become faster as we approach to the storage capacity. One possibility is that as the size of the network is $N < \infty$, the order of fluctuation in the network activity is of order $O(1/\sqrt{N})$.[8]. As we increase the number of fixed points to store, we increase the number of attractor states in the space of network states, this leads a decreasing of the size of the attractor landscape and thus the noise with respect to the size of the basins of attraction become important. As a conclusion, one can consider that the storage and retrieval of fixed points in the context of this work is possible for finite size networks knowing that this theory is valid for infinitely large networks.
Bibliography


