Analysis of Hopfield Model as Associative Memory

Silvestri Matteo

Department of Mathematics, University of Rome "La Sapienza"

Abstract

This article delves into the Hopfield neural network model, drawing inspiration from biological neural systems. The exploration begins with an overview of the model's foundations, incorporating insights from mechanical statistics to deepen our understanding. Focusing on *audio retrieval*, the study demonstrates the Hopfield model's associative memory capabilities. Through practical implementation, the network is trained to retrieve different patterns.

1 An Overview of Neuroscience

The Hopfield model finds inspiration in the intricate connections among biological neurons in the human brain, mimicking nature's efficiency in information processing. Similar to the synaptic communication in biological systems, the Hopfield network utilizes connections to store and retrieve patterns. Before delving further, we'll provide a brief overview of biological neurons, shedding light on their fundamental structures.

1.1 Biological Neurons

A neuron, the fundamental building block of our nervous system, consists of three main parts: the dendrites, the cell body (soma), and the axon. **Dendrites** receive signals from other neurons, transmitting these signals to the cell body. The **Cell body** processes these signals and, if the input is sufficient, generates an electrical impulse. This impulse travels down the **Axon**, a long, slender projection, to communicate with other neurons or muscles. **Synapses**, the junctions between neurons, facilitate this communication by transmitting electrochemical signals to other cells. This intricate architecture enables neurons to form complex networks, laying the foundation for the remarkable functionality of our nervous system.

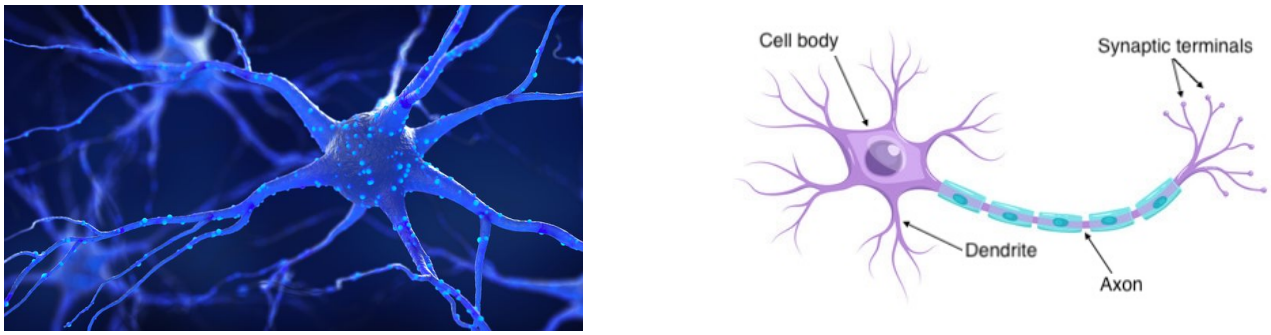


Figure 1: On the left a 3D-image of a neuron, on the right a brief summary of its structure.

Embarking on the marvels of neural networks, the sheer scale of these intricate systems is staggering. In the human brain alone, an astounding *86 billion neurons* form a vast web of

connections, orchestrating the symphony of thoughts and actions. Comparatively, the cognitive prowess of elephants shines through their expansive neural landscapes, boasting approximately 257 billion neurons—more than three times that of humans. Dolphins, renowned for their intelligence, navigate the seas with brains equipped with tens billion of neurons, contributing to their advanced problem-solving abilities. These examples illuminate the remarkable diversity and complexity of neural networks across species, and moreover show the huge quantity of neurons that are used in daily actions. In the forthcoming section, we delve into a concise overview of a neuron's behavior, unraveling the fundamental processes that underlie its intricate functioning.

1.2 Action Potential

While our focus is on the Hopfield network model, a brief foray into the neuroscience background is essential. In this section, we won't delve into the depths of neuroscience, but rather aim to illuminate a fundamental concept: *action potential*. Understanding the behavior of neurons and the process that gives rise to a spike lays a crucial foundation. This rudimentary insight serves as a key building block, enriching our comprehension of the Neural Network models and its mathematical underpinnings. The action potential $\mathcal{AP}$, a pivotal concept in neuronal function, is a neuronal phenomenon in which we see the neuron fires. Transmission of a neuronal signal is entirely dependent of the movement of ions, such as Sodium (Na+), Potassium (K+) and Chloride (CI), that are unequally distributed between the inside and the outside of the cell body. The presence and the movement of these ions creates a chemical gradient across the membrane which we define as electro-chemical gradient $\mathcal{ECG}$.

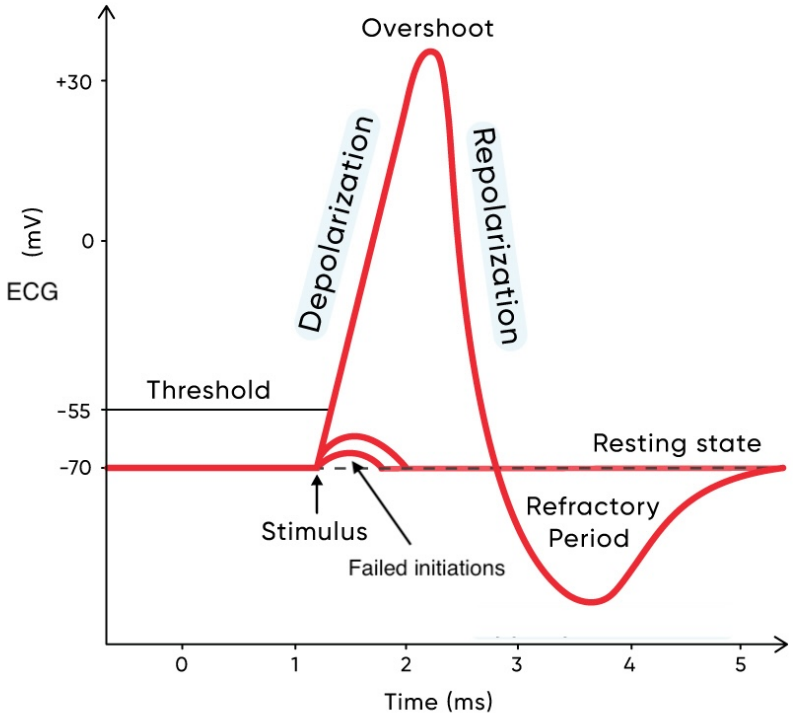


Figure 2: Action Potential

At resting state, $\mathcal{ECG}$ hovers around $-70mV$. However, when a neuron receives a stimulus, the action potential experiences a tendency to increase. Within the biological neuron, a critical threshold exists, typically around $-55mV$. If ECG exceeds this threshold, then the neuron activates and the process of generating a nerve signal begins. Otherwise, the neuron is unable to fire and tends to return to its resting state. Let's focus on the case where the **Stimulus** is strong enough to cause ECG to exceed threshold. Then the neuron activates, and the **Depolarization** process begins. In this state, the neuron begins to interact with ions present inside and outside the membrane in such a way that ECG continues to grow up to $+40mV$; this is called an overshoot. At this point, the membrane begins to expel positive ions in order to do ECG decrease; this process is called **Repolarization**. Following these processes, the neuron is able to generate an electrical signal that is sent through the axon to reach the target cell. In more detail, after the depolarization process there is the so-called **Refractory period**. During this time segment, ECG drops below the resting-state value. This happens because the channels present in the membrane that allow the ions to cross it do not close instantaneously and therefore allow values smaller than $-70mV$ to be reached. The neuron subsequently restabilizes and returns to a resting state. As we can see in figure 2 the trend of the $\mathcal{ECG}$ takes on a shape of a spike and allows us to imagine the production of an electrical signal. Navigating the intricacies of neuroscience, especially outside one's specialization, can be challenging. To enhance clarity, I've included a link to a video explanation. However, two basic concepts on which the associated mathematical models are based should be clear:

- Cognitive capacity does not depend on any intensity, but only on *binary values* (and more specifically, by frequency).
- There should exists a *threshold* of the network that allows to activate or not neurons.

2 Artificial Neurons

As we transition into the realm of "Artificial Neurons", our focus shifts from the intricate workings of biological neurons to their mathematical counterparts. These artificial neurons serve as the foundational units in computational models, mirroring the neurological properties we've explored in the preceding section. Embodying the essence of their biological counterparts, artificial neurons encapsulate key features like activation thresholds and the generation of binary outputs, all within a mathematical framework. This section unravels the fundamental principles behind these artificial neurons, bridging the gap between neuroscience and mathematical modeling in the pursuit of understanding neural networks.

2.1 McCulloch-Pitts Model

The McCulloch-Pitts model, known as **MP Neuron**, stands as the epitome of simplicity in neural network modeling. Comprising inputs, weights, and a threshold, this foundational model captures the essence of a neuron's basic functionality. Inputs convey signals, each associated with a weight, which collectively influence the neuron's behavior. The critical threshold, akin to the activation threshold in biological neurons, determines whether the neuron fires or remains at rest. As we can see from the figure 3, the model can be briefly summarized in the equation

$$y = \Theta\left(\sum_{k=1}^{N} J_k S_k - U^*\right) \tag{1}$$

where Θ is the *Heaviside function*, N is number of inputs S_k, J_k are the synaptic weights and U^* is the neuron threshold. This general model obviously embodies the main behavioral

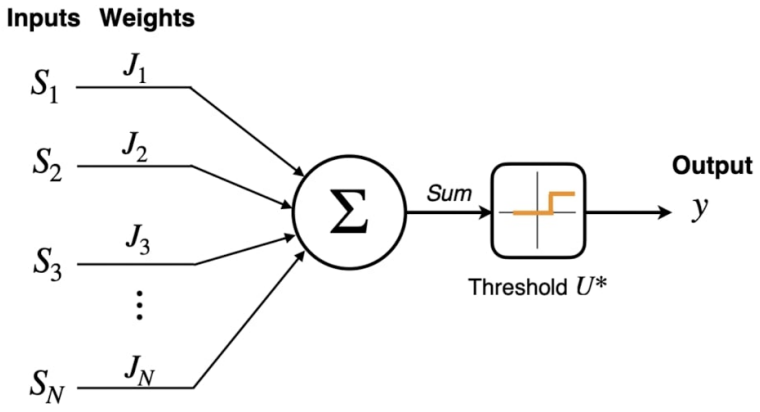


Figure 3: MP model

characteristics of biological neurons. More specifically, a network of MP neurons can perform any single-out mapping $M : \Omega \subset \{0,1\}^N \to \{0,1\}$; for instance, any boolean function of N variables, can be expressed in terms of AND,OR,NOT operations $(\wedge, \vee, \neg)$. These function can be easily implemented with a network of two MP neurons (only one in the case of $\neg$). However, if N is greater than 1, there is a counterexample that shows how a single layer of MP neurons is not sufficient to approximate any objective function M : XOR operation. For this reason, scientific attention has shifted to **MP Multilayer Networks**. Indeed, the XOR function can be performed starting from a neural network composed of two layers, each with two MP neurons. If now we assume that we don't know the function M but we have a *training set* $TS = \{x_i, M(x_i)\}_{i=1}^P \subset \Omega \times \{0,1\}$, then there exists an algorithm knows as **Perceptron** that allow us to train an MP neuron in order to emulate as best as possible the target value M(·):

Algorithm 1 Perceptron Learning

1: Define $y = y(x) = \Theta(J \cdot x - U^*)$ with randomly chosen parameters $\{J_k\}_{k=1}^N$ and U^*
2: $i = 1$
3: **while** termination condition not reached **do**
4: **if** $M(x_i) = y(x_i)$ **then** keep going
5: **if** $M(x_i) = 0 \wedge y(x_i) = 1$ **then** $U^* = U^* + 1$, $J = J - x_i$
6: **if** $M(x_i) = 1 \wedge y(x_i) = 0$ **then** $U^* = U^* - 1$, $J = J + x_i$
7: $i = i + 1$
8: **end while**

A possible termination condition is, for example, $y(x_i) = M(x_i) \quad \forall x_i \in TS$.
Therefore the algorithm ends successfully only if Ω is *linearly separable*; if not, there are methods that transform Ω into a linearly separable space in order to train the neuron correctly.

2.2 Neural Networks

From now on, we want to study the model of neural networks, in which there is mutual information between inputs and outputs. Let's consider a network with N neurons $S_1 ... S_N$; we denote with J_{ij} the synaptic weight between the neurons S_i and S_j. Since the network is no longer feed-forward, we are interested in expressing the state of each individual neuron as a function of time t. More specifically, if we assume that we know the neuron states $S_1(t) ... S_N(t)$

then the behavior of i-th neuron can be expressed by the equation

$$S_i(t+\Delta t) = \Theta\left(\sum_{k=1}^{N} J_{ik} S_k(t) - U_i^*\right) \tag{2}$$

where U_i^* is the threshold of the neuron S_i. However, this modeling turns out to be a bit unrealistic, as it does not take into account the fact that the neuron's threshold could vary over time. Indeed, we consider the **Stochastic Neurons**

$$\begin{cases} U_i^*(t) = U_i^* - \dfrac{T}{2} z_i(t) \\ S_i(t+\Delta t) = \Theta\left(\displaystyle\sum_{k=1}^{N} J_{ik} S_k(t) - U_i^*(t)\right) \end{cases} \tag{3}$$

with the *noise term* such that $\mathbb{E}\left(z_i(t)\right) = 0$ and $\mathbb{E}\left(z_i(t)^2\right) = 1$; the *temperature* T is a very important control parameter, which plays a central role in the computational and convergence properties of the Hopfield model. A second convenient traslation is to redefine neurons such that they have values in $\{+1, -1\}$, so called **Ising Neurons**

$$\begin{cases} \sigma_i(t) = 2S_i(t) - 1 \\ U_i^* = \dfrac{1}{2}\left(\displaystyle\sum_{k=1}^{N} J_{ik} - h_i\right) \end{cases} \tag{4}$$

where $\sigma_i(t) \in \{-1, +1\}$ and $\{h_i\}_{i=1}^N$ are the biases. Let's denote the *local field* acting on the neuron σ_i as

$$\varphi_i(t) := \sum_{k=1}^{N} J_{ik}\sigma_k(t) + h_i$$

After some simple calculations, we obtain the generic formula

$$\sigma_i(t+\Delta t) = \operatorname{sgn}\left(\varphi_i(t) + T z_i(t)\right) \tag{5}$$

with $\operatorname{sgn}(x) = 2\Theta(x) - 1$; this will be the notation we will use as the article continues. The probability to find a neuron state $\sigma_i(t+\Delta t)$ can be expressed in terms of the noise distribution $\mathcal{P}(z)$; in the case of *symmetric distribution*, we have

$$\mathbb{P}\left(\sigma_i(t+\Delta t) = \pm 1\right) = g\left(\pm\frac{\varphi_i(t)}{T}\right) \equiv \int_{-\infty}^{\pm\frac{\varphi_i(t)}{T}} \mathcal{P}(z)\, dz \tag{6}$$

where g is the *cumulative distribution function*. A natural choice is to consider the distribution of a Standard Gaussian, whose associated *cdf* is $g(x) = \frac{1}{2}(1 + \operatorname{erf}(\frac{x}{\sqrt{2}}))$. Another plausible choice is

$$\begin{cases} \mathcal{P}(z) = \dfrac{1}{2}(1 - \tanh^2(z)) \\ g(z) = \dfrac{1}{2}(1 + \tanh(z)) \end{cases} \tag{7}$$

As we can see from the figure 4, the two possible choices are very similar on a numerical level. Notice that T controls the impact of the noise on the model; in fact, if $T = 0$ then the process is deterministic, while if $T \to \infty$ then

$$\mathbb{P}\left(\sigma_i(t+\Delta t) = \pm 1\right) = g(0) = \frac{1}{2}$$

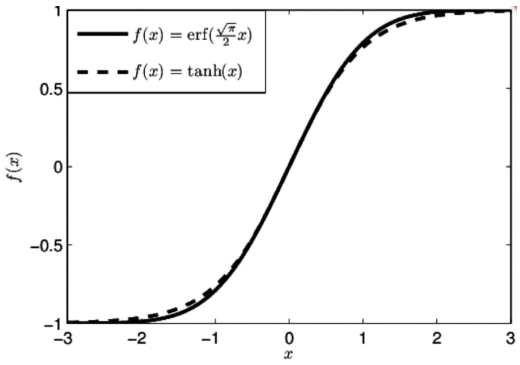


Figure 4: Comparison between cdf_s

that is, the process is fully-random. If we assume $T \neq 0$, we can see from equation 6 that the microscopic laws governing the *spin vector* $\boldsymbol{\sigma} = (\sigma_1 ... \sigma_N)$ are defined as a stochastic alignment to the local field $\boldsymbol{\varphi} = (\varphi_1(\boldsymbol{\sigma}) ... \varphi_N(\boldsymbol{\sigma}))$; as a matter of fact, if $\varphi_1(\boldsymbol{\sigma}) > 0$ then $\mathbb{P}(\sigma_1(t + \Delta t) = 1) > \frac{1}{2}$, while if $\varphi_1(\boldsymbol{\sigma}) < 0$ then $\mathbb{P}(\sigma_1(t + \Delta t) = 1) < \frac{1}{2}$.

2.3 Noiseless Networks

Now we focus on noiseless dynamics, which can be divided into two types:

- **Parallel dynamics**, represented by

$$\sigma_i(t + \Delta t) = \mathrm{sgn}\left(\sum_{k=1}^{N} J_{ik}\sigma_k(t) + h_i\right) \quad \forall i \in \{1..N\} \tag{8}$$

- **Sequential dynamics**, represented by

$$\begin{cases} \text{choose randomly } i \text{ in } \{1..N\} \\ \sigma_i(t + \Delta t) = \mathrm{sgn}\left(\sum_{k=1}^{N} J_{ik}\sigma_k(t) + h_i\right) \end{cases} \tag{9}$$

Therefore, starting from an initial configuration $\boldsymbol{\sigma}(0)$, the following sequence is obtained

$$\boldsymbol{\sigma}(0) \to \boldsymbol{\sigma}(1) \to \boldsymbol{\sigma}(2) \to ... \tag{10}$$

and we are hopeful that the sequence tends towards an *attractor* $\boldsymbol{\sigma}^*$ or in a limit cycle. Parallel dynamics and sequential dynamics (with some improvements) always evolve into an attractor, that is a limit cycle of period less than or equal to 2^N. However, we will focus on sequential dynamics since it is evident that, for very large N, parallel dynamics turns out to be very expensive from a computational point of view.

Proposition 1. *Let's consider a noiseless sequential dynamic that has the following properties:*

(i) Symmetric interactions, i.e. $J_{ik} = J_{ki} \quad \forall i, k \in \{1...N\}$

(ii) Non-negative interactions, i.e. $J_{ii} \geq 0 \ \forall i \in \{1...N\}$

(iii) Stationary external field $\boldsymbol{h} = (h_1, ..., h_N)$

Then the function

$$L(\boldsymbol{\sigma}; N, J, \boldsymbol{h}) := -\frac{1}{2}\sum_{k,l=1}^{N}\sigma_k J_{kl}\sigma_j - \sum_{k=1}^{N} h_k\sigma_k \tag{11}$$

is a ***Ljapunov function*** *with respect to the dynamic written above.*

Proof. We have to show that $\Delta L(\boldsymbol{\sigma}) = L(\boldsymbol{\sigma'}) - L(\boldsymbol{\sigma}) \leq 0 \ \forall \boldsymbol{\sigma}, \boldsymbol{\sigma'}$, where the two configurations can only be different on the state of neuron i. Wlog, we can consider $\boldsymbol{\sigma'} \neq \boldsymbol{\sigma}$ with $\sigma_i' = -\sigma_i$. Then we have

$$\begin{aligned}
\Delta L(\boldsymbol{\sigma}) &= -\frac{1}{2}\sum_{k=1}^{N} J_{ki}(\sigma_k'\sigma_i' - \sigma_k\sigma_i) - \frac{1}{2}\sum_{l=1}^{N} J_{il}(\sigma_i'\sigma_l' - \sigma_i\sigma_l) - h_i(\sigma_i' - \sigma_i) \\
&= \sum_{k=1}^{N} J_{ki}\sigma_k\sigma_i + \sum_{l=1}^{N} J_{il}\sigma_i\sigma_l - 2J_{ii} + 2h_i\sigma_i \\
&= 2\sigma_i\left(\sum_{k=1}^{N} J_{ki}\sigma_k + h_i\right) - 2J_{ii} = -2\left|\sum_{k=1}^{N} J_{ki}\sigma_k\right| - 2J_{ii} \leq 0
\end{aligned}$$

where we used the fact that $\psi\,\mathrm{sgn}(\psi) = |\psi|$ with ψ generic function; in our case, we have $\psi = \sum_{k=1}^{N} J_{ki}\sigma_k + h_i$ and $\mathrm{sgn}(\psi) = \sigma_i' = -\sigma_i$ hence the thesis. □

The result just demonstrated shows a key concept that underlies spin-glass models of neural networks and their application in patterns recognition: we want an *energy function* which has fixed points (previously called attractors) corresponding to the patterns we want the network to memorize. In the case of a **single Pattern** $\boldsymbol{\xi} = (\xi_1, ..., \xi_N) \in \{-1, +1\}^N$, we define the synaptic weights according to the *Hebbian rule*

$$J_{ij} = \frac{\xi_i\xi_j}{N} \quad \forall i, j = 1...N \tag{12}$$

which is connected to the concept "cells that fire together they wire together". Moreover, we can assume that $h_i = h \ \forall i$, because there is no reason why different neurons should be feel different external fields. Let's introduce the new variables $\tau_i = \xi_i\sigma_i$ and $\nu_i = \xi_i h$; multiplying the equation 9 by ξ_i we obtain the dynamic

$$\tau_i(t+1) = \mathrm{sgn}\left(\frac{1}{N}\sum_{k=1}^{N}\tau_k + \nu_k\right)$$

and summing over i, we can rewrite it in terms of the *average activity* $m(t) := \frac{1}{N}\sum_{i=1}^{N}\tau_i(t)$

$$m(t+1) = \frac{N_+}{N}\,\mathrm{sgn}(m(t) + |h|) + \frac{N - N_+}{N}\,\mathrm{sgn}(m(t) - |h|)$$

where N_+ is the number of positive entries of the pattern $\boldsymbol{\xi}$. This modeling fits perfectly with the retrieval task; indeed, the network have three possible behavior:

- if $m(0) > |h|$, $m(t) = 1$ then the network retrieve in 1 step the pattern $\boldsymbol{\xi}$
- if $m(0) < -|h|$, $m(t) = -1$ then we have retrieval in 1 step of the pattern $-\boldsymbol{\xi}$
- if $|m(0)| < |h|$, $\boldsymbol{\sigma}$ tends to a configuration $\boldsymbol{\xi}^0$ where all neurons have either + or − sign, depending on which sign prevails in the pattern; so $\boldsymbol{\xi}^0 = \mathrm{sgn}\left(\sum_{i=1}^{N}\xi_i\right)\mathbf{1}$.

In other words, the function m measures the alignment between the neuronal configuration $\boldsymbol{\sigma}$ and the target pattern $\boldsymbol{\xi}$. Notice that this model can reconstruct the pattern only if the initial state $\boldsymbol{\sigma}(0)$ is sufficiently close to the associated fixed point, i.e. $m(\boldsymbol{\sigma}(0)) > |h|$; otherwise, the network finds two possible configurations associated with the other two fixed points.
In the case of **multiple Patterns** $\{\boldsymbol{\xi}^\mu\}_{\mu=1}^P$ with $\boldsymbol{\xi}^\mu \in \{-1,+1\}^N$ and $P > 1$, we want a Ljapunov function that have stationary states corresponding to the patterns we want to store. For perfect retrieval, we assume that the patterns are orthogonal, i.e. $\boldsymbol{\xi}^\mu \cdot \boldsymbol{\xi}^\nu = 0 \ \forall \mu \neq \nu$; therefore, if we assume that there are no self-interactions and no external fields, Hebb's rule becomes

$$J_{ik} = \frac{1}{N}(1-\delta_{ik}) \sum_{\mu=1}^{P} \xi_i^\mu \xi_k^\mu \tag{13}$$

thus obtaining the Lyapunov function

$$L(\boldsymbol{\sigma}; N, P, J, \boldsymbol{h}=0) = \frac{P}{2} - \frac{1}{2} \sum_{\mu=1}^{P} \left[\frac{1}{\sqrt{N}} \sum_{i=1}^{N} \xi_i^\mu \sigma_i \right]$$

such that $L(\boldsymbol{\sigma}) \geq L(\pm\boldsymbol{\xi}^\mu) \ \forall \mu = 1...P$, or rather all patterns (and their opposites) are stationary points of dynamics. This will be the initial setting of the Hopfield model, which is known for its associative memory capacity with $P > 1$ patterns.

2.4 Neural Processes as Markov chains

Now, we want to analyse the previously dynamics in probabilistic terms using *Markov Chains*. In this section, we will only state the definitions and results that are of interest to us for the purpose of translating neural dynamics in terms of Markov chains.

2.4.1 Brief Overview of Markov Chains

Definition 1. *A discrete time Markov chain at values in a discrete set $\mathcal{S}$ is a sequence of random variables $X = \{X_t\}_{t\geq 1}$ such that*

$$\mathbb{P}(X_{t+1} = j | X_t = i_t, ..., X_0 = i_0) = \mathbb{P}(X_{t+1} = j | X_t = i_t) =: W_{ij}$$

i.e. that the probability of finding oneself in state j depends solely on the state at the previous time i.

The probability W_{ij} can be interpreted as the component of a *stochastic transfer matrix* W where

$$\sum_{j\in\mathcal{S}} W_{ij} = 1 \ \ \forall i \ \text{ and } \ W_{ij} \in [0,1].$$

Therefore, the Markov chain is in bi-univocal correspondence with its transfer matrix. Furthermore, we have

$$p_t(X = j | X_0) = (p_{t-1}(X = j|X_0)W)_i = ... = \left(p_0(X = j|X_0)W^t\right)_i$$

Definition 2. *A MC is said ergodic if there exists an integer τ such that for all pairs of states $(i,j) \in \mathcal{S}\times\mathcal{S}$ we have that*

$$p_t(X = j | X_0 = i) > 0 \ \ \forall t > \tau.$$

Definition 3. *A distribution $p(X|X_0)$ is invariant if $p(X|X_0)W = p(X|X_0)$.*

Theorem 1. *For any ergodic Markov chain X, there exists an unique invariant distribution $p_\infty(X|X_0)$ that is the principal left eigenvector of W, such that if $\nu(i,t)$ is the number of visits of state i in t steps then* $\lim_{t\to\infty} \frac{\nu(i,t)}{t} = p_\infty$.

Theorem 2. *Let X be an ergodic MC with invariant distribution p_∞. Then*

$$p_t(X|X_0) = p_0(X|X_0)W^t \xrightarrow{t\to\infty} p_\infty(X)$$

independently of the initial distribution $p_0(X|X_0)$.

Definition 4. *A stochastic matrix W and a measure $p(X)$ are said to be in detailed balance if*

$$p(X=i)W_{ij} = W_{ji}p(X=j) \quad \forall i,j \in \mathcal{S}.$$

2.4.2 Translating Neural dynamics in terms of Markov Chains

As analysed in section 2.2, we have that sequential dynamics with noise can be expressed by

$$\begin{cases} \text{choose randomly } i \text{ in } \{1..N\} \\ \mathbb{P}(\boldsymbol{\sigma}(t+\Delta t)) = \frac{1}{2} + \frac{1}{2}\sigma_i(t+\Delta t)\tanh(\beta\varphi_i(\boldsymbol{\sigma}(t))) \end{cases} \tag{14}$$

where $\beta = \frac{1}{T}$ and we use the fact that $\tanh(\sigma_i\beta\varphi_i(\boldsymbol{\sigma}(t))) = \sigma_i\tanh(\beta\varphi_i(\boldsymbol{\sigma}(t)))$ since σ_i takes values in $\{-1,+1\}$ and tanh is odd. If $\boldsymbol{\sigma}(t)$ is given, then equation 14 becomes

$$p_{t+1}(\boldsymbol{\sigma}) = \frac{1}{N}\sum_{i=1}^{N}\left[\left(\frac{1}{2} + \frac{1}{2}\sigma_i\tanh(\beta\varphi_i(\boldsymbol{\sigma}(t)))\right)\prod_{j\neq i}\delta_{\sigma_j,\sigma_j(t)}\right]$$

where the production tells us that all neurons remain unaffected except neuron i, while the summation tells us that during sequential dynamics, (almost) all neurons are given the opportunity to be flipped. On the other hand, if probability $p_t(\boldsymbol{\sigma})$ is given, then we get

$$p_{t+1}(\boldsymbol{\sigma}) = \sum_{\boldsymbol{\sigma'}} W[\boldsymbol{\sigma};\boldsymbol{\sigma'}]p_t(\boldsymbol{\sigma'}) \tag{15}$$

with the $2^N \times 2^N$ transfer matrix given by

$$W[\boldsymbol{\sigma};\boldsymbol{\sigma'}] = \frac{1}{N}\sum_{i=1}^{N}\left[\left(\frac{1}{2} + \frac{1}{2}\sigma_i'\tanh(\beta\varphi_i(\boldsymbol{\sigma'})\right)\delta_{\boldsymbol{\sigma},\boldsymbol{\sigma'}} + \left(\frac{1}{2} - \frac{1}{2}\sigma_i'\tanh(\beta\varphi_i(\boldsymbol{\sigma'})\right)\delta_{F_i(\boldsymbol{\sigma}),\boldsymbol{\sigma'}}\right]$$

where $F_i(\boldsymbol{\sigma}) = (\sigma_1,...,\sigma_{i-1},-\sigma_i,\sigma_{i+1},...,\sigma_N)$ is the **flipping operator**. Equation 15 is the Markov equation corresponding to the sequential process $\boldsymbol{\sigma}(t) \to \boldsymbol{\sigma}(t+1)$.

Proposition 2. *The process described above by W is ergodic. Then there exists a unique stationary distribution p_∞ to which it will converge from any initial distributions over states. This distribution is determined by the stationary condition*

$$p_\infty(\boldsymbol{\sigma}) = \sum_{\boldsymbol{\sigma'}} W[\boldsymbol{\sigma};\boldsymbol{\sigma'}]p_\infty(\boldsymbol{\sigma'}) \quad \forall\boldsymbol{\sigma} \in \{-1,+1\}^N.$$

We observe that, to calculate $p_\infty(\boldsymbol{\sigma})$, we have to solve a system of 2^N linear equations for 2^N values of p_∞, which would be a very difficult job. This is why we want to impose a strong condition that would simplify the calculations, the **Detailed Balance** :

$$W[\boldsymbol{\sigma};\boldsymbol{\sigma'}]p_\infty(\boldsymbol{\sigma'}) = W[\boldsymbol{\sigma'};\boldsymbol{\sigma}]p_\infty(\boldsymbol{\sigma}) \quad \forall\boldsymbol{\sigma},\boldsymbol{\sigma'} \in \{-1,+1\}^N. \tag{16}$$

Theorem 3. *Let's consider sequential dynamics without self-interactions ($J_{ii} = 0 \ \forall i$). Then the detailed equilibrium is equivalent to the symmetry of the interactions, i.e.*

$$J_{ij} = J_{ji} \ \ \forall i,j \iff W[\boldsymbol{\sigma};\boldsymbol{\sigma'}]p_\infty(\boldsymbol{\sigma'}) = W[\boldsymbol{\sigma'};\boldsymbol{\sigma}]p_\infty(\boldsymbol{\sigma}) \ \ \forall \boldsymbol{\sigma},\boldsymbol{\sigma'} \in \{-1,+1\}^N.$$

Moreover, if detailed balance holds then the equilibrium distribution is given by

$$p_\infty(\boldsymbol{\sigma}) \propto e^{\frac{-\mathcal{H}(\boldsymbol{\sigma})}{T}} \tag{17}$$

where $\mathcal{H}(\boldsymbol{\sigma})$ is the Ljapunov function of the noiseless dynamics given by

$$\mathcal{H}(\boldsymbol{\sigma}) = -\frac{1}{2}\sum_{i,j=1}^{N}\sigma_i J_{ij}\sigma_j - \sum_{i=1}^{N} h_i\sigma_i.$$

Notice that $p_\infty(\boldsymbol{\sigma})$ corresponds to the Boltzmann-Gibbs distribution for $(\boldsymbol{\sigma}, J, \boldsymbol{h})$.

Proof. Wlog we can suppose $\boldsymbol{\sigma'} \neq \boldsymbol{\sigma}$; moreover, we assume $\boldsymbol{\sigma'} = F_i(\boldsymbol{\sigma})$ where F_i is the flipping operator. In this case, the DB-condition is equivalent to

$$\frac{p_\infty(\boldsymbol{\sigma'})e^{-\beta\sigma_i'\varphi_i(\boldsymbol{\sigma'})}}{\cosh(\beta\varphi_i(\boldsymbol{\sigma'}))} = \frac{p_\infty(\boldsymbol{\sigma})e^{-\beta\sigma_i\varphi_i(\boldsymbol{\sigma})}}{\cosh(\beta\varphi_i(\boldsymbol{\sigma}))}$$

Notice that $\varphi_i(\boldsymbol{\sigma}) = \varphi_i(\boldsymbol{\sigma'})$ because of no self-interaction; hence the DB-condition becomes

$$p_\infty(\boldsymbol{\sigma'})e^{-\beta\sigma_i'\varphi_i(\boldsymbol{\sigma})} = p_\infty(\boldsymbol{\sigma})e^{-\beta\sigma_i'\varphi_i(\boldsymbol{\sigma})}$$

Recall that the process is ergodic, so $p_\infty(\boldsymbol{\sigma}) > 0 \ \ \forall\boldsymbol{\sigma}$; therefore, we can express the limit distrubution in terms of the exponential function

$$p_\infty(\boldsymbol{\sigma}) = \exp\left\{\beta\left(\sum_k h_k\sigma_k + \frac{1}{2}\sum_{k\neq l}\sigma_k J_{kl}\sigma_l + K(\boldsymbol{\sigma})\right)\right\}$$

where $K(\boldsymbol{\sigma})$ is the *implicit term*. Combining the two previous equations, we obtain that the DB condition equals

$$\begin{cases} \exp(g_i(\boldsymbol{\sigma'})) = \exp(g_i(\boldsymbol{\sigma})) \\ \dfrac{g_i(\boldsymbol{\sigma})}{\beta} = -\sigma_i\varphi_i(\boldsymbol{\sigma}) + \displaystyle\sum_{k=1}^{N} h_k\sigma_k + \frac{1}{2}\sum_{k\neq l} J_{kl}\sigma_k\sigma_l + K((\boldsymbol{\sigma}) \end{cases}$$

We observe that, by explicating the local field, we obtain

$$\begin{aligned}
\frac{g_i(\boldsymbol{\sigma})}{\beta} &= -\sigma_i\left(\sum_{k=1}^{N} J_{ik}\sigma_k + h_i\right) + \sum_{k=1}^{N} h_k\sigma_k + \frac{1}{2}\sum_{k\neq l} J_{kl}\sigma_k\sigma_l + K(\boldsymbol{\sigma}) \\
&= \left(-\sigma_i h_i + \sum_{k=1}^{N} h_k\sigma_k\right) + \left(-\sum_{k=1}^{N} J_{ik}\sigma_k\sigma_i + \frac{1}{2}\sum_{k\neq l} J_{kl}\sigma_k\sigma_l\right) + K(\boldsymbol{\sigma}) \\
&= \sum_{k\neq i} h_k\sigma_k + \left(-\sum_{k\neq i} J_{ik}\sigma_k\sigma_i + \frac{1}{2}\sum_{k\neq l,k\neq i,l\neq i} J_{kl}\sigma_k\sigma_l + \frac{1}{2}\sum_{k\neq i} J_{ki}\sigma_k\sigma_i + \frac{1}{2}\sum_{l\neq i} J_{il}\sigma_i\sigma_l\right) + K(\boldsymbol{\sigma}) \\
&= \{\text{terms non involving index } i\} + \frac{1}{2}\sum_{k\neq i}(J_{ki} - J_{ik})\sigma_i\sigma_k + K(\boldsymbol{\sigma})
\end{aligned}$$

In conclusion, these calculations show that the DB condition holds if and only if there exists a function $K(\cdot)$ such that $K(\boldsymbol{\sigma'}) - K(\boldsymbol{\sigma}) = \sigma_i \sum_{k\neq i}(J_{ik} - J_{ki})\sigma_k$. We can now demonstrate the two implications. Let's now assume that the balance condition applies. Thus, we consider $\boldsymbol{\sigma} = F_j(\boldsymbol{\sigma})$ with $j \neq i$ so we obtain

$$\begin{aligned} K(F_iF_j\boldsymbol{\sigma}) - K(F_j\boldsymbol{\sigma}) &= \sigma_i \sum_{k\neq i}(J_{ik} - J_{ki})F_j\sigma_k \\ &= \left(\sigma_i \sum_{k\neq i}(J_{ik} - J_{ki})\sigma_k\right) - 2\sigma_i(J_{ij} - J_{ji})\sigma_j \end{aligned}$$

whence

$$K(F_iF_j\boldsymbol{\sigma}) - K(F_j\boldsymbol{\sigma}) - K(F_i\boldsymbol{\sigma}) + K(\boldsymbol{\sigma}) = -2\sigma_i(J_{ij} - J_{ji})\sigma_j$$

and since the right-hand member is invariant with respect to permutation (i, j), the right-hand member must necessarily be invariant, i.e. there must be symmetrical interaction.
On the other hand, if we assume symmetrical interaction, then the DB condition is equivalent to the existence of a function K such that $K(F_i\boldsymbol{\sigma}) - K(\boldsymbol{\sigma}) = 0$ and this is easily verified by taking, for example, constant $K(\boldsymbol{\sigma}) = K$. □

3 Statistical Mechanics background

Statistical mechanics is a branch of Physics that elucidates the collective behavior of macroscopic systems through the analysis of statistical properties at the microscopic level.
Spin models form a foundational framework in statistical mechanics, offering a conceptual lens to investigate the collective behavior of magnetic systems. At their core, these models represent the angular momentum of atomic spins, influencing the material's magnetic properties. Notable among them is the *spin glass model*, introducing disorder for complex behaviors. A specific variant, the *mean-field spin glass*, simplifies the description for tractable analyses. These models illuminate the dynamics of magnetic materials, serving as invaluable tools to unveil the intricate interplay between microscopic spins and macroscopic magnetic phenomena. Within the domain of neural networks, spin models find a unique application in unraveling the intricate dynamics governing these complex systems. By adapting the principles of statistical mechanics to model the interactions between spins, analogous to neurons in a network, researchers can gain valuable insights into the emergent behaviors, phase transitions, and information processing mechanisms within neural networks. These spin models offer a conceptual bridge, allowing us to draw parallels between the collective behavior of spins in magnetic systems and the dynamic interactions of neurons in a network. In this interdisciplinary approach, spin models prove to be versatile tools, shedding light on the nuanced dynamics that define the computational prowess of neural networks. In this section, we provide a concise overview of key statistical mechanics concepts essential for understanding the Hopfield model. We explore principles directly relevant to this neural network, distilling the foundational elements needed to navigate the interplay between statistical mechanics and the Hopfield model's dynamics. Our aim is to offer a focused and accessible entry into the world of statistical mechanics tailored to the study of neural networks.
In general, we have a mean-field spin model with $\boldsymbol{\sigma} \in \{-1, +1\}^N$, $J \in \mathbb{R}^{N\times N}$ symmetric and the **Hamiltonian**

$$\mathcal{H}(\boldsymbol{\sigma}; N, J, \boldsymbol{h}) = -\frac{1}{2}\sum_{i,j=1}^{N} \sigma_i J_{ij}\sigma_j - \sum_{i=1}^{N} h_i\sigma_i \tag{18}$$

although our focus will be on the probability distribution

$$\rho(\boldsymbol{\sigma}; \beta, N, J, \boldsymbol{h}) = \frac{\exp\left(-\beta\mathcal{H}(\boldsymbol{\sigma}; N, J, \boldsymbol{h})\right)}{Z_{\beta,N,J,\boldsymbol{\sigma}}} \tag{19}$$

where

$$Z_{\beta,N,J,\boldsymbol{h}} = \sum_{\boldsymbol{\sigma'}} \exp\left(-\beta \mathcal{H}(\boldsymbol{\sigma'}; N, J, \boldsymbol{h})\right) \tag{20}$$

is the **Partition function**. We observe that we are exactly in the context described by Theorem 3 and this shows the close connection between associative neural network models and those of statistical mechanics. The level of description of statistical mechanics is the **Mesoscopic level**. For each *mesoscopic state* $i \in \mathcal{T}$, we consider its *energy* E_i; then, a *thermodynamic state* of the system is described by statistical set $\{\rho_i\}_{i\in\mathcal{T}}$ interpreted as a probability distribution over the set of states $\mathcal{T}$. Let us now give some definitions.

Definition 5. *For all $j = 1...|\mathcal{T}|$ we call a pure state $\rho^{(j)}$ if $\rho_i^{(j)} = \delta_{ij} \quad \forall i \in \mathcal{T}$.*

Notice that the set $\mathcal{S}$ of thermodynamic state is a simplex, so each state $\rho \in \mathcal{S}$ can be expressed as a combination of pure states.

Definition 6. *We define the following functions where k_B is the Boltzmann constant :*

(i) Internal Energy $U(E, \rho) = \sum_{i\in\mathcal{T}} \ \rho_i E_i$

(ii) Gibbs Entropy $S(\rho) = -k_B \sum_{i\in\mathcal{T}} \ \rho_i \log(\rho_i)$

(iii) ***Free Energy*** $F(E, \rho, T) = U(E, \rho) - TS(\rho)$

Definition 7. *We define a thermodynamic equilibrium,or* ***Boltzmann-Gibbs distribution****, the state $\bar{\rho}$ that minimizes the free energy F.*

Theorem 4. *It holds that*

$$\begin{cases} \bar{\rho}_i = \dfrac{e^{-\frac{E_i}{k_B T}}}{Z} \\ Z = \displaystyle\sum_{i\in\mathcal{T}} e^{-\frac{E_i}{k_B T}} = e^{-\frac{\bar{F}(E,T)}{k_B T}} \\ \bar{F}(E, T) = \inf_{\rho\in\mathcal{S}} F(E, \rho, T) = F(E, \bar{\rho}, T) \ . \end{cases}$$

Before moving on to the study of the simplest neural network model, i.e. the Curie Weiss model, let us define other functions that will be studied next.

Definition 8. *We define the following functions :*

(i) ***Intensive Free Energy*** $f_{N,\beta,J,\boldsymbol{h}} = -\frac{T}{N} log Z_{\beta,N,J,\boldsymbol{h}}$

(ii) Intensive Pressure $A_{N,\beta,J,\boldsymbol{h}} = \frac{1}{N} log Z_{\beta,N,J,\boldsymbol{h}}$

(iii) Thermal average of observable g $\quad \omega_{N,\beta,J,\boldsymbol{h}}(g) = \sum_{\boldsymbol{\sigma}} g(\boldsymbol{\sigma}) \rho_{N,\beta,J,\boldsymbol{h}}(\boldsymbol{\sigma})$

Furthermore, we will say that one of the defined functions is in the **Thermodynamic limit** (**TDL**) if we let N tend to infinity. This concept will be fundamental in the continuation of the article, as we will engage in finding solutions in this context because it would be more simple. However, the existence of this limit will not be analysed, which is a very complicated analytical problem.

3.1 Curie-Weiss Model

Let us now turn our attention to a very simple neural network model, the Curie-Weiss model. This model can be described as a system made of N spins $\sigma_i \in \{-1, +1\}$ that can interact pairwise and with an external field according to the Hamiltonian

$$\mathcal{H}_{N,J,\boldsymbol{h}}(\boldsymbol{\sigma}) = -\sum_{(i,j)} \sigma_i J_{ij} \sigma_j - \sum_{i=1}^{N} h_i \sigma_i$$

We shall consider a homogenous coupling, i.e. $J_{ij} = \frac{J}{N}$ with J constant, and homogeneous external field, i.e. $h_i = h \ \forall i = 1...N$. The *order parameter* of the model is the **empirical Magnetization**

$$m_N(\boldsymbol{\sigma}) = \frac{1}{N} \sum_{i=1}^{N} \sigma_i \tag{21}$$

which expresses the percentage of spin pointing upwards or downwards. In fact, if $m = 1$ then there is all positive spin while if $m = -1$ all negative spin. In fact, we can rewrite the Hamiltonian as a function of m as follows

$$\mathcal{H}_{N,J,\boldsymbol{h}}(\boldsymbol{\sigma}) = -\frac{J}{N} \sum_{i>j} \sum_{j} \sigma_i \sigma_j - h \sum_{i} \sigma_i = -\frac{NJ}{2} (m_N(\boldsymbol{\sigma}))^2 - hN m_N(\boldsymbol{\sigma}) + \frac{J}{2} \tag{22}$$

where the last term is the diagonal element that we added to introduce m and therefore we have to subtract it; clearly this term will be left out as it does not affect the minimisation process. This is the reason why we refer to these models as *mean field models*. This allows us to apply the so-called **Coarse-Graining** process, which consists of simplifying the expression for the partition function and rewriting it as

$$\begin{aligned} Z_N &= \sum_{\boldsymbol{\sigma}} e^{-\beta \mathcal{H}_N(\boldsymbol{\sigma})} = \sum_{m \in \mathcal{M}} e^{-\beta \mathcal{H}_N(m) \Omega_N(m)} \\ &\approx \sum_{m \in \mathcal{M}} e^{-\beta F_N(m)} \approx \sum_{k} e^{-\beta N f_N(m_k^*)} \rho_N^{(k)} \end{aligned}$$

where $\Omega_N(m)$ is the number of configurations with magnetization equal to m and m^* is the argmin of the function $[U_N(m) - TS_N(m)]$. The approximations have been made assuming that Ω_N is equal to Gibbs entropy (and not Boltzmann entropy) and that N is very large. Now we want to derive an explicit formula for free energy via **Laplace's method**.

Theorem 5. *Let $g \in \mathcal{C}^2([a,b])$ and $x_0 \in (a,b)$ be the only point such that $g(x_0) = \max_{x \in [a,b]} g(x)$ and $g''(x_0) < 0$. Then*

$$\lim_{N \to \infty} \frac{\int_a^b e^{Ng(x)} \, dx}{e^{Ng(x_0)} \sqrt{\frac{2\pi}{N(-g''(x_0))}}} = 1. \tag{23}$$

This holds also for $a = -\infty$ and/or $b = +\infty$ and for $x \in \mathbb{R}^K$ with $\lim_{N \to \infty} \frac{K}{N} = 0$.

Corollary 1. *Let be g a convex function. Then we have*

$$\lim_{N \to \infty} -\frac{1}{N} \log \int_{-\infty}^{+\infty} dx \ e^{Ng(x)} = \min_x g(x). \tag{24}$$

We observe that we can use the theorem to rewrite the distribution function as

$$\begin{aligned} Z_N &= \int dm\ Z_N(m) = \int dm\ e^{-N\beta f_N(m)} \\ &\overset{N>>1}{\approx} \int dm\ e^{-N\beta f_N(m) e^{-\frac{N\beta}{2} f_N''(m^*)(m-m^*)^2}} \\ &= e^{-N\beta f_N(m^*)} \sqrt{\frac{2\pi}{N\beta f_N''(m^*)}} \end{aligned}$$

from which we obtain that

$$f_N = -\frac{1}{N\beta} log Z_N \overset{N>>1}{\approx} f_N(m^*) + \frac{log\left(\frac{2\pi}{\beta f''(m^*)}\right) - log N}{2N\beta} \overset{N\to\infty}{\longrightarrow} f(m^*).$$

Thus we have obtained that, under TDL, the intensive free energy is simply $f(m)$ calculated at its minimum m^*, which in turn provides the expectation of the magnetization; so we want an explicit formula for f. Notice that

$$f(m) = -\lim_{N\to\infty} \frac{T}{N} log Z_N(m) = \frac{1}{2} J m^2 - hm - Ts(m) \tag{25}$$

where $s(m) = \lim_{N\to\infty} \frac{1}{N} log \Omega_N(m)$ is the entropy per spin. We will now calculate $s(m)$ using the integral representation of Dirac's delta. Indeed, we have

$$\begin{aligned} \delta(m - m_N(\boldsymbol{\sigma}) &= \int_{-\infty}^{+\infty} dx\ \frac{N}{2\pi} e^{iNx[m - m_N(\boldsymbol{\sigma})]} = \int_{-\infty}^{+\infty} dx\ \frac{N}{2\pi} e^{iNxm - ix\sum_j \sigma_j} \\ &= \int_{-\infty}^{+\infty} dx\ \frac{N}{2\pi} e^{iNxm} \prod_{j=1}^{N} e^{-i\sigma_i x} \end{aligned}$$

thus

$$\begin{aligned} \Omega_N(m) &= \sum_{\boldsymbol{\sigma}} \delta(m - m_N(\boldsymbol{\sigma}) = \sum_{\boldsymbol{\sigma}} \int_{-\infty}^{+\infty} dx\ \frac{N}{2\pi} e^{iNxm} \prod_{j=1}^{N} e^{-i\sigma_i x} \\ &= \int_{-\infty}^{+\infty} dx\ \frac{N}{2\pi} e^{iNxm} \prod_{j=1}^{N} \sum_{\sigma_j = \pm 1} e^{-i\sigma_i x} \\ &= \int_{-\infty}^{+\infty} dx\ \frac{N}{2\pi} e^{iNxm} \left(2\cos(x)\right)^N = \int_{-\infty}^{+\infty} dx\ \frac{N}{2\pi} e^{iNxm} e^{N\log(2cos(x))}. \end{aligned}$$

From Corollary 1, we get

$$\frac{d}{dx}[imx - \log(2\cos(x))] = im - \tan(x) = 0 \iff x^* = \operatorname{atan}(im) = i\operatorname{atanh}(m) = \frac{i}{2}\log\left(\frac{1+m}{1-m}\right)$$

from which

$$s(m) = -m\tanh(m) + \log(2\cosh(\tanh(m))) = -\frac{1+m}{2}\log\left(\frac{1+m}{2}\right) - \frac{1-m}{2}\log\left(\frac{1-m}{2}\right)$$

where where we put $m = x^*$. In conclusion, we obtain the explicit formula for the **coarse-grained intensive free energy in TDL** given by

$$f(m) = -\frac{J}{2}m^2 - hm + T\left[\frac{1+m}{2}\log\left(\frac{1+m}{2}\right) + \frac{1-m}{2}\log\left(\frac{1-m}{2}\right)\right] \tag{26}$$

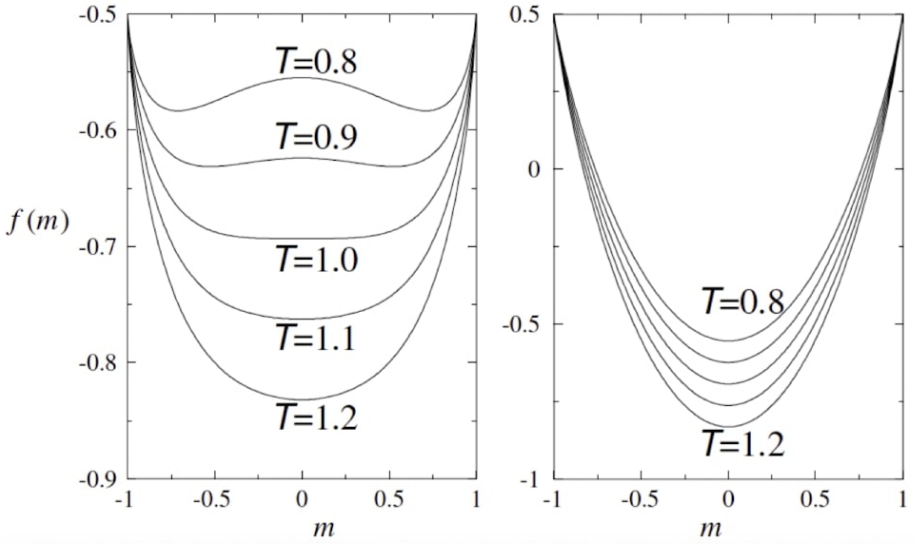


Figure 5: Coarse-grained free energies for the case $J = 1$ (left) and $J = -1$ (right)

where the first two addends correspond to the energy contribution, while the third addend is the entropy contribution governed by the control parameter T. Figure 5 shows the plot of $f(m)$ at different temperature levels. We observe that if J is less than zero, i.e. we are in the presence of *paramagnetic* material, the coarse-grained free energy has a single minimum at $m = 0$. Whereas if J is greater than zero, i.e. there is *ferromagnetic* behaviour, then the minimum points vary as the temperature varies. This is a direct consequence of the fact that

$$f'(m) = 0 \implies Jm^*\beta + \beta h = \text{atanh}(m^*) = 0 \implies m^* = \tanh(\beta(Jm^* + h)) \tag{27}$$

so the minima correspond to the solutions of the **Self-Consistency equation** which clearly vary as β varies. See Appendix A for further details.

3.2 Phase Transitions and Ergodicity Breaking

A *phase transition* occurs when there is a singularity in the free energy or in one of its derivative. This aspect is related to a sharp change in the properties of a substance: for example liquid/gas or paramagnetic/ferromagnetic. It is possible to classify these transitions. For istance, if there is a finite discontinuity in one or more of the prime derivatives, we will say that we are in the presence of a first-order phase transition; if, on the other hand, the prime derivatives are continuous but the second derivatives are discontinuous or finite, then the transition will be of the second order. Phase transitions often involves a **Symmetry Breaking process**; in fact, the Hamiltonian's system and the equations of the dynamics are invariant under the action of a symmetry group, but the system is not. Tipically, the high-temperature phase contains more symmetries than the low-temperature phase. We denote by T_c a critical temperature associated with a phase transition which is sensitively to the interatomic interactions. When $T > T_c$, then the free energy has a single minimum at $m^* = 0$ and thus the system explores the entire admissible phase space. If $T < T_c$, then the free energy has two symmetric minima $\pm m^* \neq 0$ and the order of the system is restricted to an appropriate part of the phase space Ω^+ or Ω^- with $\Omega = \Omega^+ \cup \Omega^-$. In the first case the system is ergodic, while in the second case ergodicity is broken. If T=0, the system just moves toward configuration leading to energy minimization so also in this case ergodicity is broken. But in section 2.4.2 we said that the neural dynamics is ergodic; so what is wrong? The fact is that in the TDL $N \to \infty$ the energy barriers cannot

be overcome and then ergodicity is broken. It is possible to see this from the point of view of Markov chains; in particular, we denote with $\rho_j(t)$ as the probability that the system is in state j at time t and $\boldsymbol{\rho} = (\rho_1 ... \rho_M)$ where $M = 2^N$ as all the possible states. Then we can express the probability of being at time $t+1$ in state i as

$$\rho_i(t+1) = \sum_{j=1}^{M} W(i \mid j)\rho_j(t).$$

with $\boldsymbol{W}$ transition matrix. Let us consider its the spectral expansion

$$W(i \mid j) = \sum_{k=1}^{M} \lambda_k v_k^L(j) v_k^R(i)$$

from which we obtain

$$\rho_i(t) = \boldsymbol{W}^t \rho_i(0) = \sum_{j=1}^{M}\sum_{k=1}^{M} \lambda_k^t v_k^L(j) v_k^R(i) \rho_j(0).$$

Recall that left and right eigenvectors are orthogonal if they belong to different eigenvalues.

Theorem 6. *Let $\boldsymbol{W}$ be a stochastic matrix. Then*

(i) $|\lambda| \leq 1$ *for all* $\lambda \in \sigma(\boldsymbol{W})$

(ii) $\lambda_{max} = 1$ *and* $\boldsymbol{v}^L = (1, \ldots, 1)$

This implies that

$$\rho_i(t) \xrightarrow{t\to\infty} \sum_{j=1}^{M} v_{k^*}^R(i)\rho_j(0) = v_{\lambda_{\max}}^R(i)$$

then , after a long enough time, memory of the initial state is lost and , for any initial configuration, the asymptotic distribution is reached corresponding to the right eigenvector of the unique largest eigenvalue. Thus, as long as N is finite and $T > 0$, the system is ergodic.

So how is it possible to break the ergodicity? As M increases, the spectral gap between the maximum and minimum eigenvalue may decrease; then in the limit $M \to \infty$, i.e. in the TDL, there may be an asymptotic degeneration and ergodicity is broken. For istance, let us denote $v_{1,2}^R, v_{1,2}^L$ the right and left eigenvector associated to the max eigenvalue with degeneration 2. Then we obtain

$$\rho_i(t) \xrightarrow{t\to\infty} \sum_{k=1}^{2}\sum_{j=1}^{M} v_k^L(j) v_k^R(i)\rho_j(0) = \sum_{k=1}^{2} [v_k^L \boldsymbol{\rho}(0)] v_k^R$$

and since $\boldsymbol{\rho}(0)$ still appears in the equation, the process will lead to different asymptotic trajectories depending on the initial condition.

The Curie Weiss model, although we have comprehensive knowledge, is a very simple neural network model. In fact, we have seen how the cost function (intensive free energy or the Hamiltonian) has, at best, three points of minimum: $m = 0$ in correspondence of a completely disordered pattern, $m = \pm m^*$ in correspondence of a $\pm\boldsymbol{\xi}$ pattern. Thus the model can store with N neurons only one pattern, which is why we will now analyse in detail the much better performing *Hopfield model*.

4 Hopfield Model

Let us now analyse a more complex model. We want to obtain a neural network consisting of N neurons, which is characterised by a cost function representing P local minima at the patterns that we want to store in the network. Indeed, let us consider the patterns

$$\boldsymbol{\xi}^\mu = (\xi_1^\mu, ..., \xi_N^\mu) \in \{-1, +1\}^N \quad \forall \mu = 1...P.$$

Let us consider the standard Hamiltonian of equation 11, where now Hebb's rule is given by

$$J_{ij} = \sum_{\mu=1}^{P} \frac{\xi_i^\mu \xi_j^\mu}{N} \quad \forall i, j = 1...N. \tag{28}$$

As we did for the Curie-Weiss model, we rewrite the Hamiltonian as a function of magnetization. In this case, however, we do not use the average magnetization but the so-called **Mattis Magnetization** expressed by

$$m_{N,\mu}(\boldsymbol{\sigma}; \boldsymbol{\xi}) = \frac{1}{N} \sum_{i=1}^{N} \xi_i^\mu \sigma_i \quad \forall \mu = 1...P \tag{29}$$

i.e. the vector $(m_{N,1}, ..., m_{N,P})$. We observe that m represents the percentage of equal spins between the $\boldsymbol{\sigma}$ configuration and the $\boldsymbol{\xi}^\mu$ pattern; indeed, if $m_{N,\mu}(\boldsymbol{\sigma}) = 1$ this means that the configuration is exactly identical to the stored pattern. If we assume that we have no external field, then we get

$$\mathcal{H}_{N,P,\boldsymbol{\xi}}(\boldsymbol{\sigma}) = -\frac{1}{2} \sum_{(i,j)} \sum_{\mu=1}^{P} \frac{\xi_i^\mu \xi_j^\mu}{N} \sigma_i \sigma_j = -\frac{N}{2} \sum_{\mu=1}^{P} (m_{N,\mu}(\boldsymbol{\sigma}))^2 + \frac{P}{2} \tag{30}$$

where the second addend, which we shall ignore, is linked to the diagonal term. In the remainder of the chapter, we will analyse the solution of the Hopfield model in two separate cases:

- *Low-load* case, where P is finite (will be the case we will implement in Python)
- *High-load* case, where $P \propto N$ and $\lim_{N\to\infty} \frac{P}{N} > 0$

4.1 Solution in the Low-load Regime

Using Laplace's method, we want to obtain an explicit expression for the free energy. In more detail, we analyse the so-called **Quenched Intensive Free-energy**

$$f^Q_{N,\beta,J} = -\frac{1}{\beta N} \mathbb{E}[\log Z_{N,\beta,\boldsymbol{\xi}}] \tag{31}$$

where the average is a *quenched average* over possible realisations of patterns given by

$$\mathbb{E}[\,\cdot\,] = 2^{-NP} \sum_{\boldsymbol{\xi} \in \{-1,+1\}^{N\times P}} [\,\cdot\,]$$

although, to lighten the notation, we will avoid emphasising that all the functions we are now going to calculate are quenched. We observe that the partition function can be written as

$$\begin{aligned}
Z_{N,\beta,\boldsymbol{\xi}} &= \sum_{\boldsymbol{\sigma}} \exp\left(\frac{\beta}{2N}\sum_{i,j,\mu}\xi_i^\mu\xi_j^\mu\sigma_i\sigma_j\right) = \sum_{\boldsymbol{\sigma}}\int\left[\prod_{\mu=1}^{P} dm_\mu \delta(m_\mu - \sum_i \frac{\xi_i^\mu\sigma_i}{N})\right]\exp\left(\frac{\beta}{2N}\sum_{\mu=1}^{P} m_\mu^2\right) \\
&= \sum_{\boldsymbol{\sigma}}\int\int\left(\prod_{\mu=1}^{P} dm_\mu \frac{N}{2\pi} d\tilde{m}_\mu\right)\exp\left(iN\sum_\mu \tilde{m}_\mu m_\mu - i\sum_{j,\mu}\tilde{m}_\mu\xi_j^\mu\sigma_j + \frac{\beta}{2N}\sum_\mu m_\mu^2\right) \\
&= \sum_{\boldsymbol{\sigma}}\int\int\left(\prod_{\mu=1}^{P} dm_\mu \frac{N}{2\pi} d\tilde{m}_\mu\right)\exp\left[iN\sum_\mu \tilde{m}_\mu m_\mu + \sum_j \log\left(2\cos\left(\sum_\mu \tilde{m}_\mu\xi_j^\mu\right)\right) + \frac{\beta}{2N}\sum_\mu m_\mu^2\right]
\end{aligned}$$

so that the partition function has a linear spin-dependency and can therefore sum directly over configurations. Notice that the extremality conditions in order to use Laplace's method are

$$\tilde{m}_\mu = i\beta m_\mu \;\text{ with respect to }\; m_\mu \;\text{ and }\; m_\mu^* = m_\mu = \frac{i}{N}\sum_j \xi_j^\mu \tanh\big(\tilde{m}_\mu\xi_j^\mu\big) \;\text{ wrt to }\; \tilde{m}_\mu$$

from which we obtain

$$f_{\beta,\boldsymbol{\xi}} = \min_{\boldsymbol{m}}\left[\frac{1}{2}\sum_\mu m_\mu^2 - \frac{1}{\beta}\mathbb{E}\left(\log\left(2\cosh\beta\sum_\mu m_\mu\boldsymbol{\xi}^\mu\right)\right)\right]$$

where we use the fact that, for all $\xi_j \in \{-1,+1\}^P$

$$\frac{1}{N}\sum_j \log\left(2\cos\left(\sum_\mu \tilde{m}_\mu\xi_j^\mu\right)\right) = \frac{1}{N}\sum_j g(\xi_j) \overset{N\to\infty}{\longrightarrow} \mathbb{E}[g(\xi_j)].$$

Theorem 7. *The quenched free energy of the Hopfield model in the thermodynamic limit and in low-load regime is*

$$f_\beta^Q = \frac{1}{2}(m_\mu^*)^2 - \frac{1}{\beta}\mathbb{E}\left[\log 2\cosh\left(\beta\sum_\mu m_\mu^*\boldsymbol{\xi}^\mu\right)\right] \tag{32}$$

where the Mattis magnetization satisfy the self-consistency equations

$$m_\mu^* = \mathbb{E}\left[\boldsymbol{\xi}^\mu\tanh\left(\beta\sum_\nu m_\nu^*\boldsymbol{\xi}^\nu\right)\right]. \tag{33}$$

We can see from equation 33 how the Curie-Weiss model is a special case of the Hopfield model. In fact, if we assume that only one pattern is the candidate to be retrieved, for istance $\boldsymbol{\xi}^1$, then we have $m_1 \neq 0$ while $m_\mu = 0 \;\; \forall\mu > 1$ so

$$m_\mu^* = \mathbb{E}\left[\boldsymbol{\xi}^\mu\tanh\big(\beta m_1^*\boldsymbol{\xi}^1\big)\right] = \tanh(\beta m_1^*)\;\mathbb{E}\left[\boldsymbol{\xi}^\mu\boldsymbol{\xi}^1\right] = \tanh(\beta m_1^*)\delta_{\mu,1}$$

which is exactly the Curie-Weiss law. Also in this case, there exists a critical $\beta_c = 1$ such that if $\beta < \beta_c$ then there is a paramagnetic behavior and if $\beta > \beta_c$ then there is a ferromagnetic behavior. The most important difference between the Curie-weiss model and the Hopfield model in low-load regime is the possibility of encountering the system in a **Spurious state**: it can be interpreted as a system error during the retrieval process, where $\boldsymbol{m}^{(n)} = (1...1, 0...0)$ i.e. the magnetization is a vector with the first n components equal to 1 and the remaining equal to 0.

This solution is compatible with equation 33; in fact, if $\mu > n$, then the operator $\mathbb{E}$ factorizes because the argument of $\tanh(\cdot)$ is independent of ξ^μ so $\mathbb{E}[\xi^\mu] \cdot \mathbb{E}[\tanh \beta \sum_\nu m_\nu^{(n)} \xi^\nu] = 0$; while if $\mu \leq n$, then the equation has non-zero solution for $\beta > 1$. However, the solution associated with a spurious state is simply a free energy extremal point and not a global minimum point. In more detail, deriving an explicit expression of the Hessiana derivative of f_β, we obtain that if n is odd then there exists $T_c^{(n)}$ such that for $T < T_c^{(n)}$ the spurious states $\boldsymbol{m}^{(n)}$ are local minima, while for n even there are saddle points.

4.2 Signal-to-noise Analysis

We now ask whether Hebb's rule stabilises the stored pattern $\boldsymbol{\xi}^\mu$. For example, will the configuration $\boldsymbol{\sigma}$ given by $\sigma_i = \xi_i^\mu \ \ \forall i = 1...N$ be dynamically stable? If we assume the absence of external fields and noise, the stability condition is equivalent to saying that $\sigma_i \varphi_i(\boldsymbol{\sigma}) > 0 \ \ \forall i = 1...N$. In this way, the configuration does not vary during neural dynamics according to eq.5, hence the pattern is a fixed point for the model. For istance, let's consider $\boldsymbol{\sigma} = \boldsymbol{\xi}^1$; recall that

$$\varphi_i(\boldsymbol{\sigma}) = \sum_{j=1}^{N} J_{ij}\sigma_j(t) = \frac{1}{N}\sum_{j\neq i}\sum_{\mu=1}^{P} \xi_i^\mu \xi_j^\mu \sigma_j$$

thus we obtain

$$\sigma_i \varphi_i(\boldsymbol{\sigma}) = \xi_i^1 \varphi_i(\boldsymbol{\xi}^1) = \frac{N-1}{N} + \frac{1}{N}\sum_{j\neq i}\sum_{\mu=2}^{P} \xi_i^1 \xi_i^\mu \xi_j^\mu \xi_j^1 \tag{34}$$

where the first addend is associated with the *signal* and the second with *noise*. We observe that the signal term is equal to 1 in the TDL, while the noise term for very large N, denoted by R, verifies

$$|R| \overset{N>>1}{\approx} \sqrt{\frac{P}{N}}$$

so if P is finite and N very large, the noise becomes negligible in relation to the signal and thus each pattern is effectively a fixed point. This result remains valid even if a finite fraction of spins is flipped away at random from one of the patterns. Although our aim was to build a model and its cost function in such a way as to have minima in correspondence with patterns, the non-linearity of the dynamics means that additional attractors are created. Indeed, let's consider a configuration given by

$$\sigma_i^{(3)} = \mathrm{sgn}(\xi_i^1 + \xi_i^2 + \xi_i^3) \ \ \forall i = 1...N$$

whose Mattis magnetization equals

$$m_\mu^{(3)} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{sgn}(\xi_i^1 + \xi_i^2 + \xi_i^3)\xi_i^\mu \overset{N>>1}{\approx} \mathbb{E}[\mathrm{sgn}(\xi_i^1 + \xi_i^2 + \xi_i^3)\xi_i^\mu] = \frac{1}{2} \ \ \forall \mu = 1,2,3$$

and $m_\mu^{(3)} = 0 \ \ \forall \mu > 3$. Then we have

$$\sigma_i^{(3)} \varphi_i(\boldsymbol{\sigma}^{(3)}) = \sigma_i^{(3)} \sum_{\mu=1}^{3} m_\mu^{(3)} \xi_i^\mu + \frac{1}{N}\sum_{j=1}^{N}\sum_{\mu>3} \sigma_i^{(3)} \xi_i^\mu \xi_j^\mu \sigma_j^{(3)}$$

and we can establish that this configuration is also stable since, for very large N, we have the first term $S = \frac{1}{2}|\xi_i^1 + \xi_i^2 + \xi_i^3|$ and the second term R such that $R^2 \approx \frac{P-3}{N}$ so as before, the noise term is negligible. Now, we want to use these results to obtain a **Statistical Estimate**

of the Storage, i.e. the number of patterns $P_c = \max\{P \text{ s.t. we have retrieval}\} = \alpha N$.
We observe that the S,R terms in equation 34 verify

$$S \overset{N\to\infty}{\longrightarrow} 1 \ , \ R \overset{N\to\infty}{\longrightarrow} \mathcal{N}(0,\alpha)$$

thus we obtain

$$\mathbb{P}[\sigma_i = \xi_i^1 \text{ stable}] = \mathbb{P}[\xi_i^1 \varphi_i(\boldsymbol{\xi}^1) > 0] = \mathbb{P}[R > -1] = \frac{1}{2}\left[1 + \text{erf}(\frac{1}{2\alpha})\right].$$

If we assume $\alpha << 1$, then we can approximate the error function as $\text{erf}(x) \overset{x>>1}{\approx} 1 - \frac{\exp(-x^2)}{\sqrt{x\pi}}$ thus achieving that

$$\mathbb{P}[\boldsymbol{\sigma} = \boldsymbol{\xi}^1 \text{ stable}] \approx \left[1 - \sqrt{\frac{\alpha}{2\pi}}\exp^{-\frac{1}{2\alpha}}\right]^N \approx 1 - N\sqrt{\frac{\alpha}{2\pi}}\exp^{-\frac{1}{2\alpha}} = 1 - N_{\text{err}}.$$

So let us impose the condition $N_{\text{err}} << 1$, which is satisfied for $\alpha = \frac{1}{2\log N}$. In conclusion, the critical number of patterns to guarantee stability of pattern $\boldsymbol{\xi}^1$ is

$$P_c = \frac{N}{2\log N} \tag{35}$$

and if we want stability also for the other patterns, one reaches $P_c = \frac{N}{4\log N}$. In the particular case of the model implementation that we shall see in Chapter 5, we have $N = 513$ and thus $P_c \approx 0.1$.

4.3 Solution in the High-load Regime

In this section, we will analyse the model in the case of *High-load*, where $P \propto N$ and

$$\alpha := \lim_{N\to\infty} \frac{P}{N} > 0. \tag{36}$$

One possible approach is to use the so-called **Replica trick**. This method uses the following identity

$$\log(x) = \lim_{n\to 0} \frac{x^n - 1}{n}$$

in order to calculate the quenched pressure

$$A_\beta^Q = -\beta f_\beta^Q = \lim_{N\to\infty}\lim_{n\to 0} \frac{\mathbb{E}Z_{N,\beta,J}^n - 1}{Nn}. \tag{37}$$

The trick is to consider distinct replicas $\boldsymbol{\sigma}^{(a)}$,$\boldsymbol{\sigma}^{(b)}$ that have the same initial distribution. This leads us to introduce the new order parameter called **overlap**

$$q_{ab} = \frac{1}{N}\sum_{i=1}^N \sigma_i^{(a)}\sigma_i^{(b)} \tag{38}$$

which measures the correlation between the two replicas. This method is very efficient for solving the *Sherrington-Kirkpatrick model*, however it is more complicated to apply it to the Hopfield model (see appendix C for more informations about the SK model).
An alternative and much more sophisticated approach is the so-called **Intepolation technique**. The main idea is to introduce an interpolating pressure $A_N(t)$ that recovers the original

model for $t = 1$, while for $t = 0$ it corresponds to the pressure of a simpler model analytically addressable. Then, the expression for $A_N(t)$ is obtained by exploiting the fundamental theorem of calculus

$$A^{\mathcal{Q}}_{N,\beta} = A_N(1) = A_N(0) + \int_0^1 \frac{d}{dt}\, A_N(t')\, dt'. \tag{39}$$

The resolution is based on two starting assumptions. The first consists of the so-called **Replica Symmetry Ansatz**, in which it is assumed that $q_{ab} = q\ \forall a \neq b$. The second, consists of considering the patterns in the following way: the target pattern $\boldsymbol{\xi}^1$ is a Rademarcher pattern, while all others $\{\boldsymbol{\xi}^\mu\}_{\mu=2\dots P}$ are distributed as a standard Gaussian. Recall that the partition function is

$$Z_{N,\beta,\boldsymbol{\xi}} = \sum_{\boldsymbol{\sigma}} \exp(-\beta \mathcal{H}_{N,\beta,\boldsymbol{xi}}(\boldsymbol{\sigma})) \quad = \sum_{\boldsymbol{\sigma}} \exp\left(\frac{\beta}{2N} \sum_{i,j} \xi^1_i \xi^1_j \sigma_i \sigma_j + \frac{\beta}{2N} \sum_{\mu>1} \sum_{i,j} \xi^\mu_i \xi^\mu_j \sigma_i \sigma_j \right)$$

and using the identity

$$\int dz \exp\bigl(-Az^2 + Bz\bigr) = \sqrt{\frac{\pi}{A}} \exp\left(\frac{B^2}{4A}\right) \tag{40}$$

on the second term with $A = \frac{1}{2}$ and $B = \frac{\beta}{N} \left(\sum_i \xi^\mu_i \sigma_i\right)^2$ we obtain

$$Z_{N,\beta,\boldsymbol{\xi}} = \sum_{\boldsymbol{\sigma}} \int d\mu(z) \exp\left(\frac{\beta}{2N} \sum_{i,j} \xi^1_i \xi^1_j \sigma_i \sigma_j + \sqrt{\frac{\beta}{N}} \sum_{\mu>1} \sum_{i} \xi^\mu_i \sigma_i z_\mu \right)$$

where $\mu(z)$ is the Gaussian measure and z_μ is a real variable. The orders parameters shall be the overlap q_{12}, the Mattis magnetization m_1 and also

$$r_{12} = \frac{1}{P-1} \sum_\mu z^{(1)}_\mu z^{(2)}_\mu \ ,\ r_{11} = \frac{1}{P-1} \sum_\mu z^{(1)}_\mu z^{(1)}_\mu.$$

Now, we will simply illustrate the results required to arrive at the solution of the Hopfield model, leaving the very tiring calculations to the reader. Let be $t \in \mathbb{R}^+$, A, B, C, D constants to be set a posteriori and $J_i, \tilde{J}_\mu \sim \mathcal{N}(0,1)$; then the partition function is

$$\begin{aligned} Z_{N,\beta,\boldsymbol{\xi}}(t; J, \tilde{J}) = \sum_{\boldsymbol{\sigma}} \int d\mu(z) \exp \Bigg\{ & \frac{t}{2} \beta N m_1^2 + \sqrt{\frac{t\beta}{N}} \sum_{\mu,i} \xi^\mu_i \sigma_i z_\mu \\ & + (1-t) N D m_1 + (1-t) \frac{C}{2} \sum_\mu z^2_\mu \\ & + \sqrt{1-t} A \sum_i J_i \sigma_i + \sqrt{1-t} B \sum_\mu \tilde{J}_\mu z_\mu \Bigg\} \end{aligned} \tag{41}$$

and the corresponding quenched statistical pressure is

$$A_N(t) = \frac{1}{N} \mathbb{E} \log Z_{N,\beta,\boldsymbol{\xi}}(t) \ ,\ A(t) = \lim_{N\to\infty} A_N(t).$$

Recall that the $\langle \cdot \rangle$ average of the observable O is

$$\langle O \rangle = \mathbb{E}[\omega_{N,\beta,J}(O)] = \mathbb{E}\left[\frac{\sum_{\boldsymbol{\sigma}} O(\boldsymbol{\sigma}) \exp(-\beta \mathcal{H}(\boldsymbol{\sigma}))}{\sum_{\boldsymbol{\sigma}} \exp(-\beta \mathcal{H}(\boldsymbol{\sigma}))} \right]. \tag{42}$$

Proposition 3. *The derivative of the quenched statistical pressure at finite N is*

$$\frac{d}{dt}A_N(t) = \frac{\beta}{2}\langle m_1^2\rangle + \frac{\beta P}{2N}(\langle r_{11}\rangle - \langle r_{12}q_{12}\rangle) - D\langle m_1\rangle - \frac{A^2}{2}(1-\langle q_{12}\rangle) - \frac{\beta^2}{2}(\langle r_{11}\rangle - \langle r_{12}\rangle) - \frac{C}{2}\langle r_{11}\rangle$$

and in the thermodynamic limit we have

$$\frac{d}{dt}A(t) = \frac{\beta^2}{2}\overline{m} - \frac{\beta\alpha}{2}\overline{r}(1-\overline{q})$$

with some fixed $\overline{m}, \overline{q}, \overline{r}$ and $A = \beta\alpha\overline{r}, B = \beta\overline{q}, C = \beta(1-\overline{q}), D = \beta\overline{m}$.

Theorem 8. *In the TDL and under RS assumption, the quenched statistical pressure of the Hopfiel model is*

$$\begin{aligned} A_{\beta,\alpha}(\boldsymbol{m}, q, r) = &-\frac{\beta}{2}\boldsymbol{m}^2 + \log 2 - \frac{\alpha\beta}{2} - \beta\alpha\overline{r}(1-\overline{q}) + \frac{\beta\alpha\overline{q}}{2(1-\beta(1-\overline{q}))} \\ &- \frac{\alpha}{2}\log(1-\beta(1-\overline{q})) + \mathbb{E}\left[\log 2\cosh\left(\beta\overline{m} + J\sqrt{\beta\alpha\overline{r}}\right)\right] \end{aligned} \tag{43}$$

where $\overline{m}, \overline{q}, \overline{r}$ fulfill the conditions

$$\begin{cases} \overline{q} = \mathbb{E}\left[\tanh^2(\beta\overline{m} + J\sqrt{\beta\alpha\overline{r}})\right] \\ \overline{m} = \mathbb{E}\left[\tanh\left(\beta\overline{m} + J\sqrt{\beta\alpha\overline{r}}\right)\right] \\ \overline{r} = \dfrac{\beta\overline{q}}{(1-\beta(1-\overline{q}))^2} \end{cases}$$

In conclusion, the free energy of the system is given by $f_{\beta,\alpha}(\boldsymbol{m}, q) = -\frac{A_{\beta,\alpha}(\boldsymbol{m},q)}{\beta}$ and the extremality conditions are

$$\begin{aligned} \boldsymbol{m} &= \int d\mu(z)\mathbb{E}\left\{\boldsymbol{\xi}\tanh\left[\beta\left(\boldsymbol{m}\cdot\boldsymbol{\xi} + \frac{\sqrt{\alpha q}}{1-\beta(1-q)}z\right)\right]\right\} \\ q &= \int d\mu(z)\mathbb{E}\left\{\boldsymbol{\xi}\tanh^2\left[\beta\left(\boldsymbol{m}\cdot\boldsymbol{\xi} + \frac{\sqrt{\alpha q}}{1-\beta(1-q)}z\right)\right]\right\}. \end{aligned} \tag{44}$$

See Appendix A for some graphic results.

4.4 Phase Diagram

We now have all the tools to analyse the phase diagram of the Hopfield model in detail. As can be seen from the figure 6, four different states can be verified:

- *Retrieval state*, i.e. $\langle m\rangle \neq 0$, $\langle q\rangle \neq 0$.
- *Retrieval Spurious state*, i.e. $\langle m\rangle \neq 0$, $\langle q\rangle \neq 0$.
- *Spin-glass state*, i.e.$\langle m\rangle = 0$, $\langle q\rangle \neq 0$.
- *Paramagnetic state*, i.e.$\langle m\rangle = 0$, $\langle q\rangle = 0$.

Firstly, we observe that the retrieval state is characterised by a non-zero magnetization, while the non-retrieval state is characterised by a null, which means that the final configuration is completely random. Secondly, the Retrieval state may be perfect and thus have overlap equal to 1, or it may be in a Spurious state and retrieve only part of the pattern, thus having the

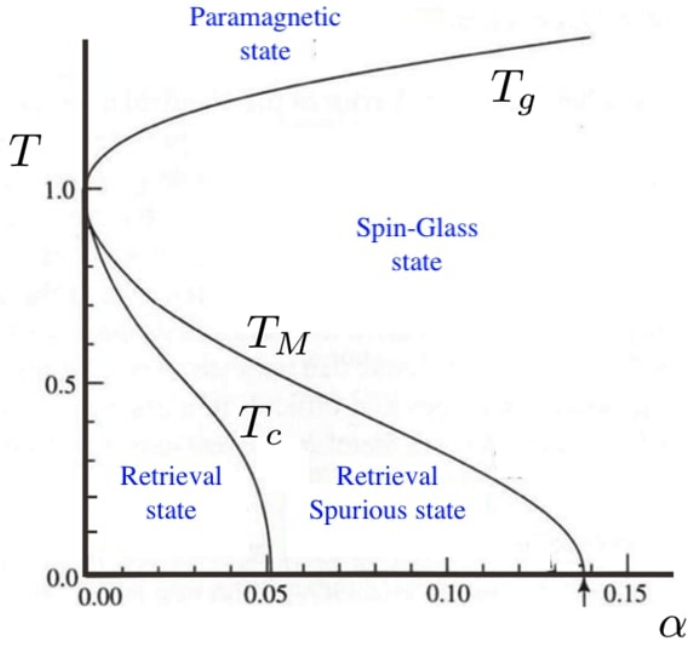


Figure 6: Phase diagram of the Hopfield model at high-load

overlap between different replicas be less than 1. However, this subdivision is more important from the point of view of model applications than from an analytical point of view.. The separation lines between the different states were derived by analytically solving the model for each grid point in order to divide the space (α, T) into the four regions. In more detail, we have that

$$T_g = 1 + \sqrt{\alpha} \tag{45}$$

while T_c and T_M were derived numerically by comparing the free energies of the pure states and those of the spin-glass states for the same α value.

5 Audio Retrieval

A practical application of the Hopfield model to a real dataset is now illustrated. The dataset consists of 81 voice recordings, in which all numbers from 0 to 80 are said. The format of these recordings is ".wav" and the Code 1 in **Python** was used to transform voice patterns into binary patterns for the network to store. We note that the *Fourier transform* was used to reduce the dimensionality of the audio data; the parameters used (n-fft,hop-length) were set to $(1024, 512)$ to maintain good audio quality and are powers of two because this provides an enormous computational advantage. Subsequently, an average was taken over each time instant so that a numerical vector of length 513 representing our audio data could be obtained. In the figure 7 the vector of the number 65 is represented. The second function of the algorithm creates a data structure in which the true audio signal with its sampling rate, matrix and mean vector of the Fourier transform is saved for each audio date. We can clearly see that the vector of coefficients is centred in 0, so the third function takes care of transforming the vector into a pattern with values in $+1, -1$ so that it can be handled by the Hopfield network. This binarization method, although very crude, turns out to be sufficient for audio retrieval. However, the patterns do not turn out to be orthogonal: in fact, on average, each pattern has

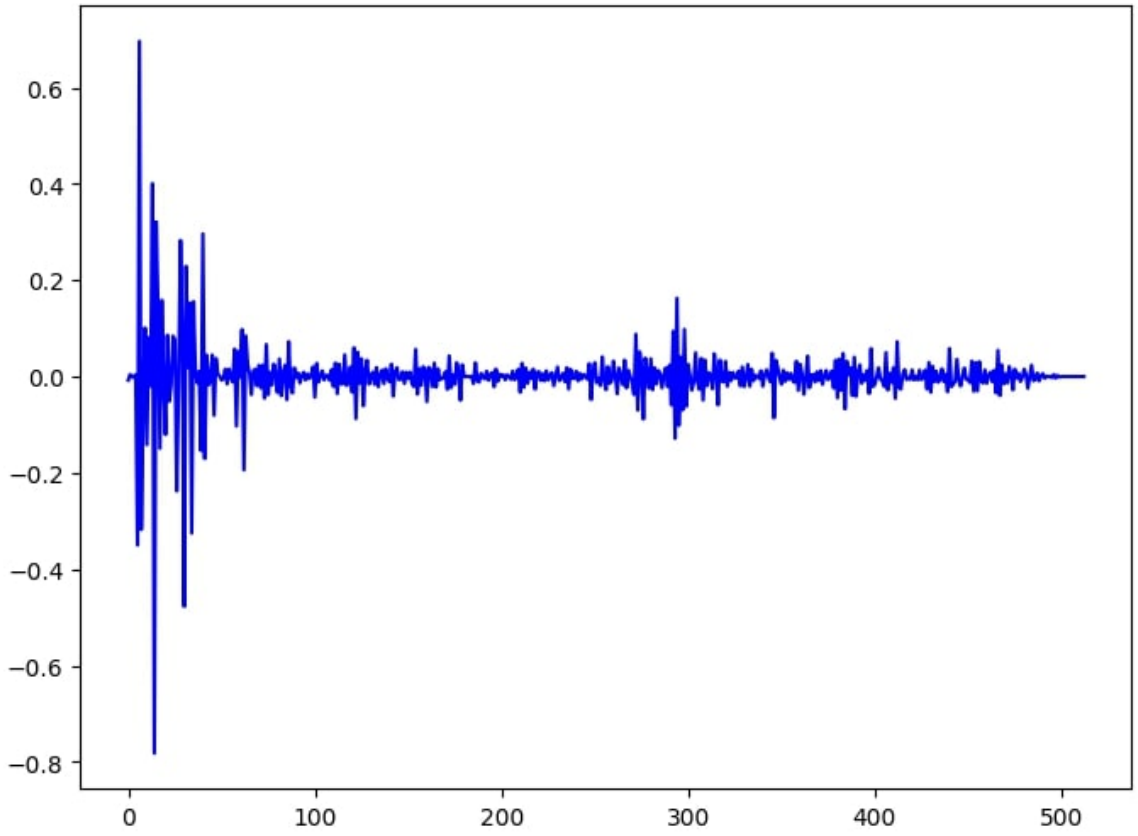


Figure 7: Representation of the average vector of Fourier coefficients of the number 65

```python
import numpy as np
import librosa

def fft_signal(audio_path):
    audio, sr = librosa.load(audio_path)
    stft_signal = librosa.stft(audio, n_fft=1024, hop_length=512)
    stft_coeff = np.mean(stft_signal, axis=1)
    return audio, sr, stft_signal, stft_coeff

def audio_importation():
    audio_objects = []
    for i in range(80):
        file_path = f'/Users/silver22/registrazioni/{i}.wav'
        audio, sr, stft_signal, stft_coeff = fft_signal(file_path)
        audio_object = {'audio':audio,'sr':sr,'stft_signal': \
 stft_signal,'stft_coeff':stft_coeff}
        audio_objects.append(audio_object)
    return audio_objects

def audio_binarization(stft_coeff):
    binary = (stft_coeff > 0).astype(int)
    binary = 2 * binary - 1
    return binary
```

Listing 1: Dataset construction

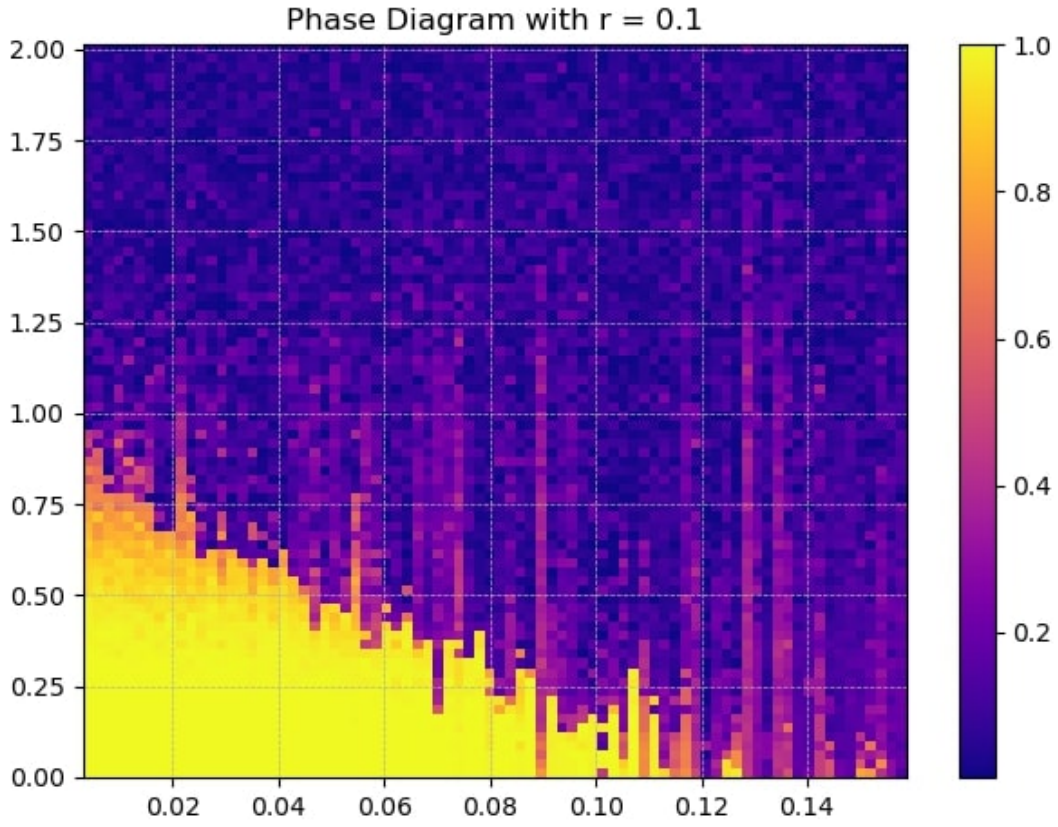
```
import numpy as np
from tqdm import tqdm
import random

class HopfieldNetwork(object):

    def train_weights(self, train_data):
        print("Start to train weights...")
        self.num_neuron = train_data.shape[1]
        self.num_patterns = train_data.shape[0]
        self.patterns = train_data
        J = np.zeros((self.num_neuron, self.num_neuron))
        for i in tqdm(range(0, self.num_neuron)):
            for j in range(i + 1, self.num_neuron):
                for mu in range(0, self.num_patterns):
                    J[i, j] += train_data[mu, i] * \
  train_data[mu, j]
        J = (J + J.T) / self.num_neuron
        self.J = J

    def predict(self, test_data, temperature):
        sigma = test_data.copy().T
        sigma = sigma.T
        N=self.num_neuron
        K=self.num_patterns
        alpha = K/N
        T = temperature
        beta = 1.0 / T
        MCstat_step=50
        MCrelax_step=1
        magn_mattis_matrix = np.zeros((self.num_patterns, \
  MCstat_step))
        for stat in range(0, MCstat_step):
            for step in range(0,MCrelax_step):
                for i in range(0,N):
                    k = np.random.randint(0, N)
                    deltaE=2*sigma[k]*np.dot(sigma,self.J[:,k])
                    ratio=np.exp(-beta*deltaE)
                    gamma=np.minimum(ratio,1)
                    if np.any(random.uniform(0,1) < gamma):
                        sigma[k] = -sigma[k]  #flipping
            for mu in range(0,self.num_patterns):
                magn_mattis_matrix[mu,stat]= np.dot(sigma, \
 self.patterns[mu,:])/N
        predicted = sigma
        return predicted,magn_mattis_matrix
```

Listing 2: Hopfield Network implementation

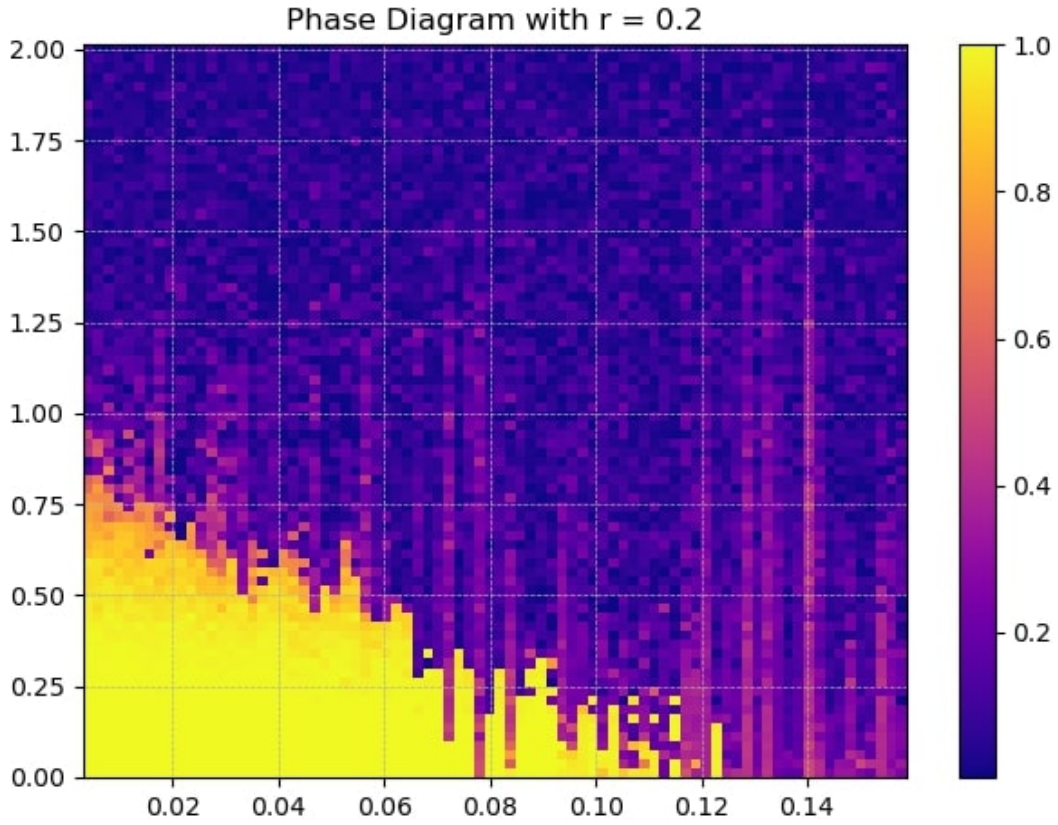
50% of the components equal to the other patterns. Let us now move on to the construction of the model. As we can see from Code 6, the Hopfield model can be implemented with two simple functions: the first (train-weights) takes as input a matrix containing the patterns to be stored as rows, and creates the J matrix of synaptic weights according to Hebb's rule. The second (predict), simulates the process of equation 10 using *Monte Carlo simulations* (see appendix B for further details). Now, let us analyse the performance of the pattern by doing the following test: we randomly take a pattern from among those stored, and corrupt it by inverting a percentage of components equal to a randomness r. We then feed this configuration to the model and compare the Mattis magnetisation relative to that specific pattern. Clearly, we will have that the model has successfully retrieved if m is about 1, while the prediction has failed if m is less than 0.5. We will use this heuristic to create graphs that play the same role as the phase diagram, i.e. we will say that we are in a

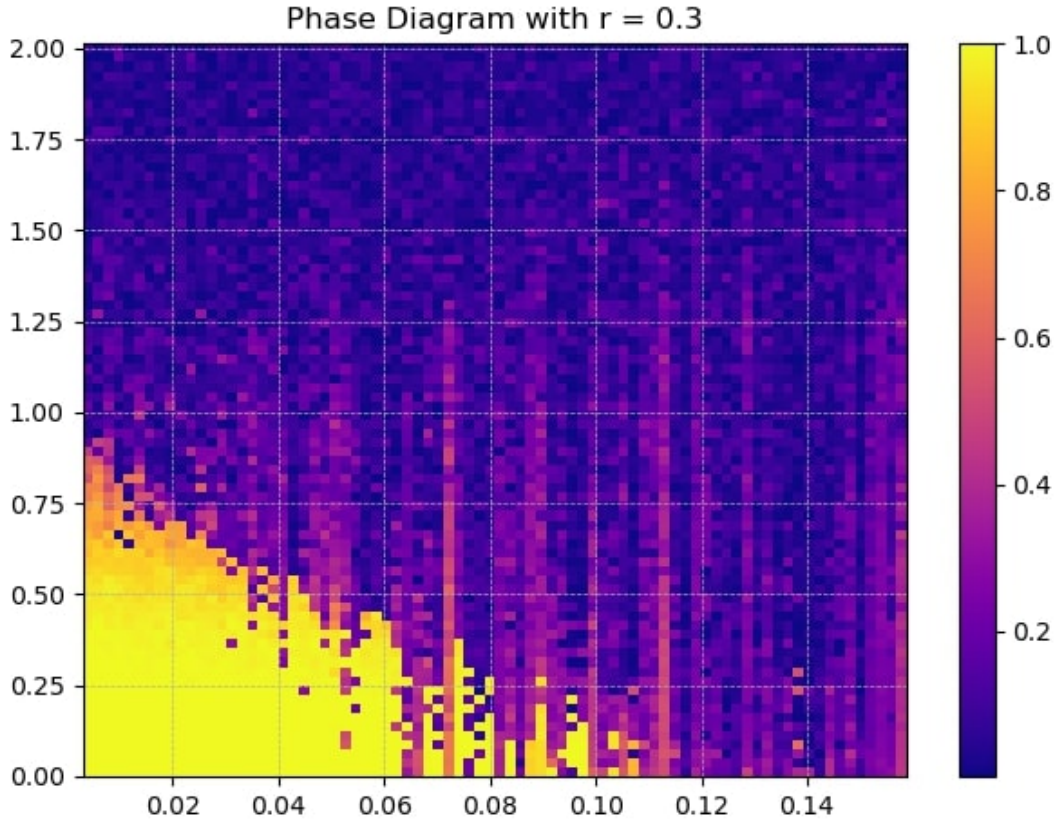
- Retrieval state if m_μ is greater than 0.9
- Spurious state if m_μ is between 0.6 and 0.9
- non-Retrieval state if m_μ is less than 0.6.

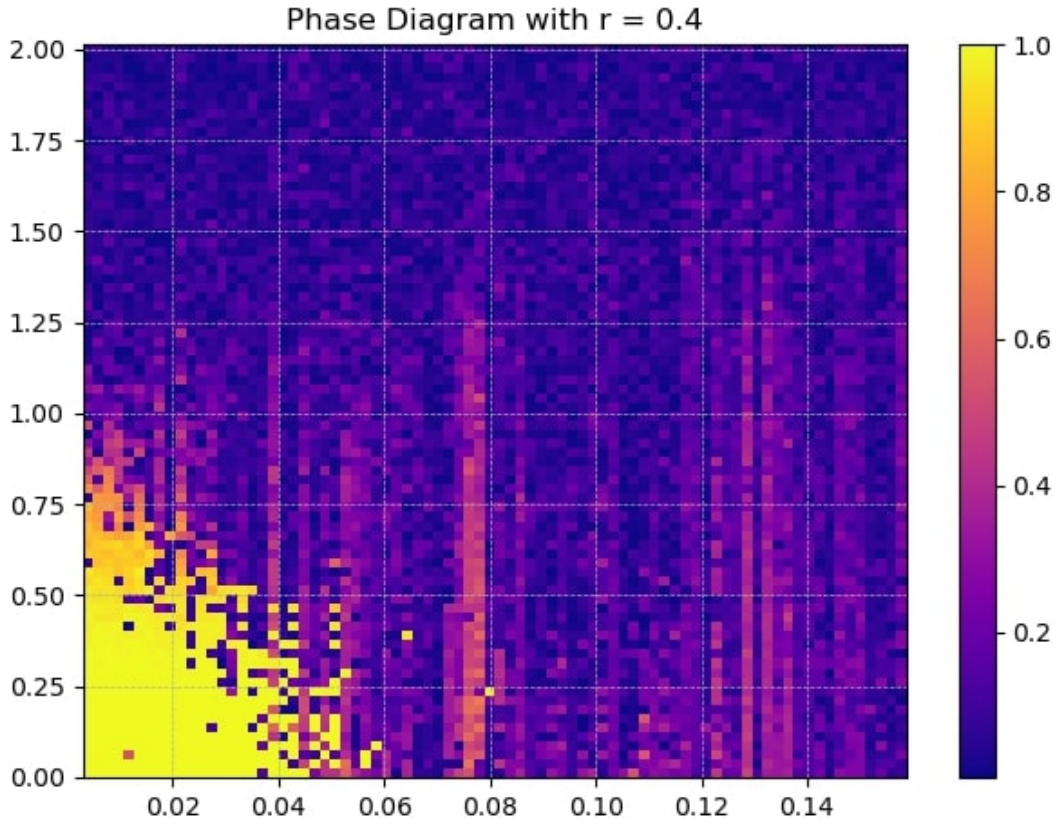
In more detail, we will analyse this process iteratively. That is, at each step a pattern is added to the model and tested; in this way, we want to test the performance of the model as the load changes. In order to make the test as generic as possible, and considering that the voice patterns that make up the dataset have a strong correlation between them (think, for example, of the numbers $21, 22$, etc...), we will perform this pattern addition randomly. At each step, i.e. at a fixed number of patterns, the model is made to work with a temperature ranging from 0.01 to 2. In the colourplots shown in the figure, the magnetization value at the end of the Monte Carlo simulation is used. In particular, we show the phase diagram with respect to different randomness values with which to corrupt the test data.

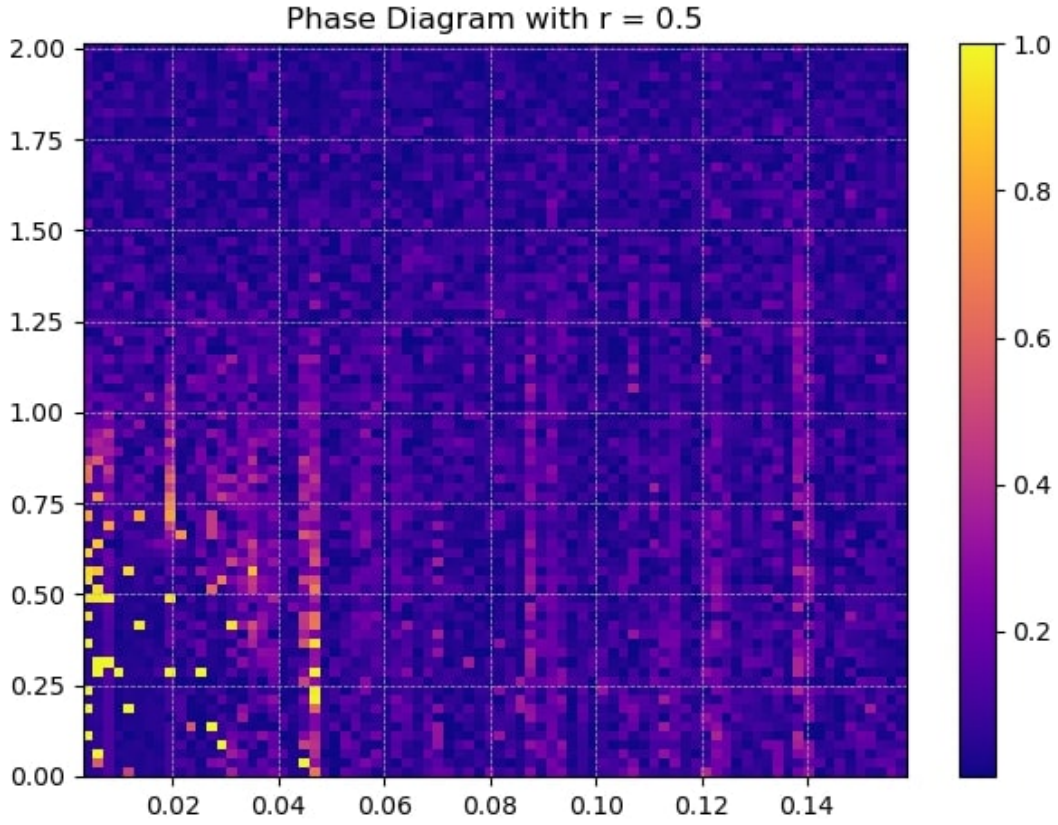


Below you can find the Code 3 that calculates the following graphs.


Phase Diagram with r = 0.2
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
1.0
0.8
0.6
0.4
0.2


Phase Diagram with r = 0.3
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
1.0
0.8
0.6
0.4
0.2


Phase Diagram with r = 0.4
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
1.0
0.8
0.6
0.4
0.2


Phase Diagram with r = 0.5
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
1.0
0.8
0.6
0.4
0.2

```
import numpy as np
import matplotlib.pyplot as plt
from tqdm import tqdm
import random
def get_corrupted(self, pattern,r):
        sample_size = int(self.num_neuron*r)
        I = np.random.choice(len(pattern),size = sample_size, \
 replace=False)
        corrupted = pattern.copy()
        for i in range(len(I)):
            corrupted[I[i]] = -1*corrupted[I[i]]
        return corrupted

audio_object = audio_importation()
num_patterns = 81
I = np.arange(81)
np.random.shuffle(I)
I_new = I.copy()
patterns_bin_total_list = []
for iter in range(2, num_patterns+1):
    binary_list=[]
    patterns_bin = np.zeros((iter,len(audio_object[0] \
['stft_coeff'])))
    for i in range(0,iter):
        stft_coeff = audio_object[I_new[i]]['stft_coeff']
        binary = audiobin2.audio_binarization(stft_coeff)
        patterns_bin[i,:] = binary
    patterns_bin_total_list.append(patterns_bin)

model = HopfieldNetwork()
T = T = np.linspace(0.01, 2, 80)
A = np.zeros(80)
for i in range(0,80):
    A[i] = (i+2)/513
magns = np.zeros((len(T),len(A)))
for iter in range(0,num_patterns-1):
    model.train_weights(patterns_bin_total_list[iter])
    print("We're using",iter+2, "patterns")
    rand_test = np.random.choice(range(model.num_patterns))
    randomness = 0.2
    test = patterns_bin[rand_test,:]
    test_corrupted = get_corrupted(test,randomness)
    for t in tqdm(range(0,len(T))) :
        predicted,magnetization = model.predict(test_corrupted, \
 temperature=T[t])
        magns[t,iter] = np.abs(magnetization[rand_test,-1])

plt.pcolormesh(A, T, magns, cmap='plasma')
plt.colorbar()
plt.grid(True, linestyle='dashed', linewidth=0.5)
plt.title('Phase Diagram')
plt.tight_layout()
plt.show()
```

Listing 3: Creation of one colourplot with r=0.2

6 Conclusion

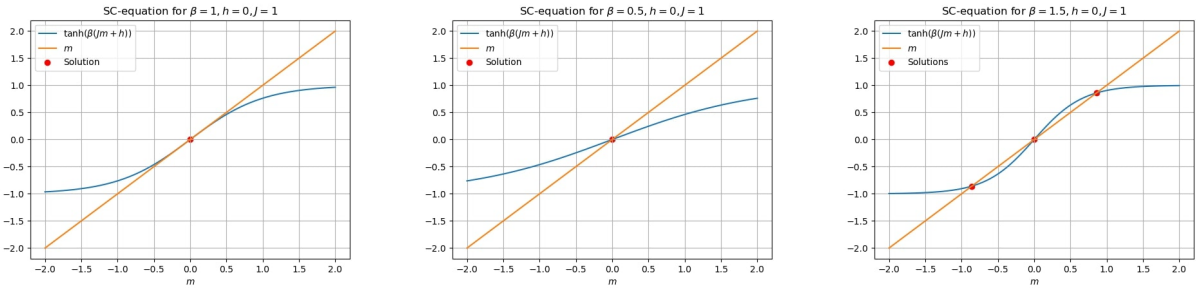
In this last section we want to draw conclusions about the results obtained in the previous chapter. In particular, we want to highlight the limitations and strengths of the Hopfield model. Firstly, we can say that this model achieves astonishing results compared with its simplicity from both an implementation and a numerical point of view. However, the fact that one must necessarily work with patterns with values in $\{-1, +1\}$ can be a problem in real life, as the binarization of real data may result in a significant loss of the information contained by them. In our case, it was possible to work well with audio data thanks to the Fourier transform, but this may not be the case with other types of data. Secondly, the 'empirical' phase diagrams obtained in Chapter 5 and the theoretical one obtained in Chapter 4 show a property of the model that is also common to the biological brain: there is a maximum limit of information that the network can store while maintaining good performance. This can be a great limitation in today's world, since the amount of data is increasingly large and therefore it would be computationally unsustainable to store an NxN matrix with $N >> 1$. Thirdly, we have seen from the various graphs in Chapter 5 that the model can recognise corrupted patterns up to a certain threshold of randomness; this means that if the network encounters a pattern that is excessively corrupted, it will not be possible to recover the original pattern. This may also be a limitation of the model, because there are currently other neural network models that are able to clean a data item from noise more efficiently (e.g. Autoencoders). Following the analysis of these model limitations, further models much more complex than Hopfield's are already being developed, such as Dense Associative Memories. In my humble opinion, these neural network models are fascinating and will continue to be in the spotlight of many researchers around the world, as, unlike almost all Deep Learning models, they have a very detailed mathematical background that allows for a theoretical analysis of the model's capabilities. Having a theory behind a model is very advantageous from a practical as well as an economic point of view, since it would be possible to choose the hyperparameters a priori and not having to test the model using many GPUs. In conclusion, the Hopfield model performs well as an associative memory model for image or audio data and is capable of recovering significantly corrupted patterns.

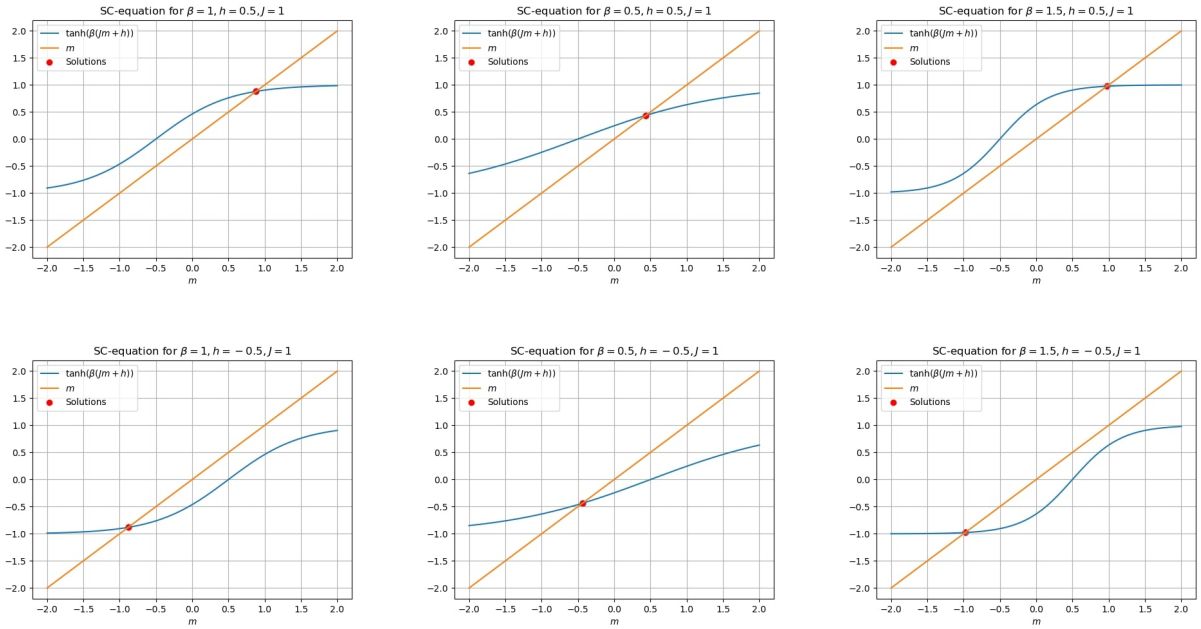
Appendix A: Self-Consistency Equations

Let us analyse the self-consistency equations that characterise the calculation of the extreme points of free energy in the case of the Curie-Weiss model. The equation is

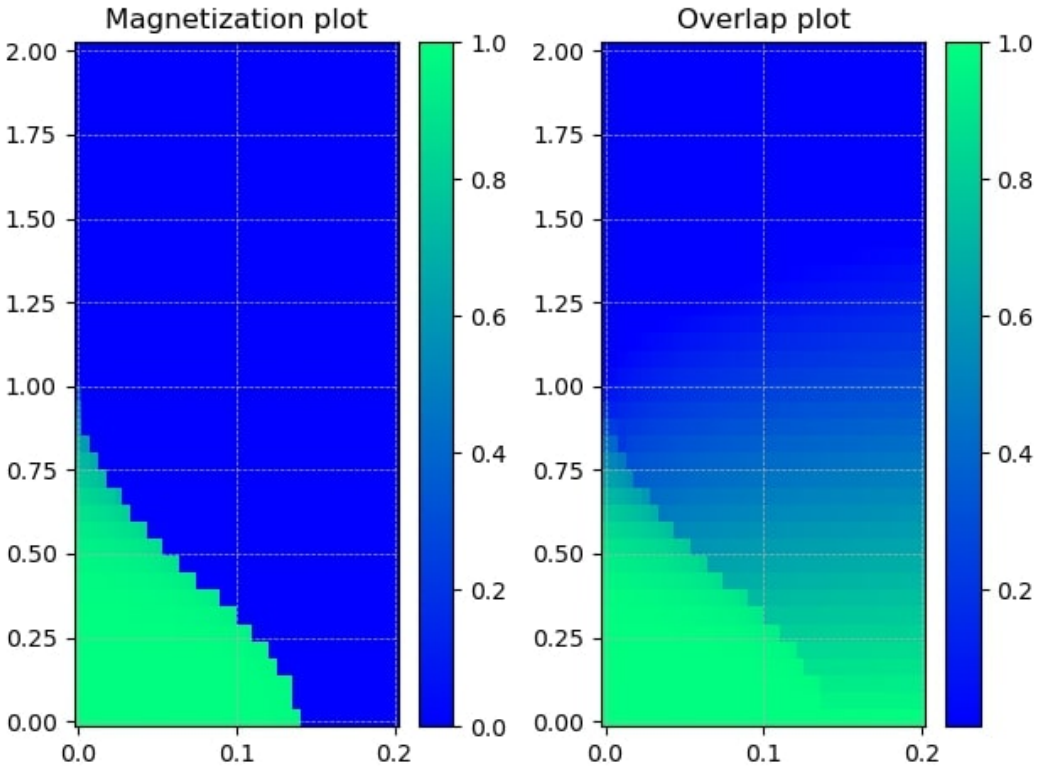
$$m = \tanh[\beta(Jm + h)]$$

and we show the solutions as the parameters β, J, h vary using the fixed point method. Notice that if $\beta > 1$ then $T < 1$ and we have more solutions, while if $\beta \leq 1$ we have a unique solution.





Below is also the graphical solution of the self-consistency equations of the high-load hopfield model with the RS assumption using Equation 44 from which we can explicitly see the Retrieval zone and the Spin-Glass zone.



Appendix B: Monte Carlo Simulations

We explain step by step the Monte Carlo simulation used in the *predict* function in Code 6. We observe that the following process was used to perform sequential dynamics while optimising the computational cost.. Let's consider the simulation

```
        sigma = test_data.copy().T
        sigma = sigma.T
        N=self.num_neuron
        K=self.num_patterns
        alpha = K/N
        T = temperature
        beta = 1.0 / T
        MCstat_step=50
        MCrelax_step=1
        for stat in range(0, MCstat_step):
            for step in range(0,MCrelax_step):
                for i in range(0,N):
                    k = np.random.randint(0, N)
                    deltaE=2*sigma[k]*np.dot(sigma,self.J[:,k])
                    ratio=np.exp(-beta*deltaE)
                    gamma=np.minimum(ratio,1)
                    if np.any(random.uniform(0,1) < gamma):
                        sigma[k] = -sigma[k]
```

The simulation hyper-parameters are $MCstat-step$ and $MCrelax-step$. The former indicates the number of steps to be taken before considering the final value as a statistic. The second, indicates how much you want to relax the process, i.e. how many steps you have to take so that the distribution associated with the network state is close to the Boltzmann-Gibbs distribution, which therefore minimises the free energy. These parameters must be chosen a posteriori, testing the convergence of the magnetisation as the parameters change. In our code, we decided not to relax the process and it turns out that 50 Monte Carlo steps are sufficient for the model to have convergence in the case of retrieval. The third for loop with respect to variable i corresponds to a single simulation step. The variable i takes values between 1 and N in such a way that all neurons in the network can be inverted according to the following strategy. At each i-step, a neuron is randomly chosen; then the energy contribution $deltaE$ associated with that neuron is calculated. If $deltaE$ is less than 0, for the minus in the Hamiltonian this gives a positive contribution and thus the state of the neuron will be flipped with probability 1. If $deltaE$ is greater than 0, then the probability of finding the neuron in that state is calculated according to the usual formula associated with the Boltzmann-Gibbs distribution; then a random number is extracted in $(0, 1)$ and the state of the neuron is inverted only if the number extracted is smaller than the probability calculated previously. This gives a chance to reverse the state of the neuron even though this new configuration does not lead to a decrease in energy. The sigma configuration that will emerge from these three chained cycles will correspond to the new configuration that will hopefully be equal to some fixed point that has been stored in the network.

Appendix C : Sherrington-Kirkpatrick Model

The Hamiltonian of the model is

$$\mathcal{H}^{SK}_{N,J} = -\frac{1}{2\sqrt{N}} \sum_{i,j} J_{ij}\sigma_i\sigma_j$$

where the synaptic weights are distributed as a standard Gaussian, i.e. $J_{ij} \sim \mathcal{N}(0,1)$. Notice that the Hamiltonian depends on some random parameters whose probability is supposed to be known, which is why the model is well suited to analysing *disordered systems*. Normalisation with the square root is motivated by the fact that this gives $\langle \mathcal{H} \rangle \propto N$. In the remainder of the appendix, we report the solution of the model using the *Replica Trick* with the *Replica Symmetric Ansatz* (see equation 37). We have

$$\begin{aligned}
\mathbb{E}[Z^n_{N,\beta,J}] &= \mathbb{E}\left[\sum_{\boldsymbol{\sigma}^{(1)}} ... \sum_{\boldsymbol{\sigma}^{(n)}} \exp\left(-\beta \sum_{a=1}^{n} \mathcal{H}_{N,\beta,J}(\boldsymbol{\sigma}^{(a)})\right)\right] \\
&= \mathbb{E}\left[\sum_{\boldsymbol{\sigma}^{(1)}...\boldsymbol{\sigma}^{(n)}} \exp\left(\frac{\beta}{2\sqrt{N}} \sum_{a=1}^{n} \sum_{(i,j)} J_{ij}\sigma_i^{(a)}\sigma_j^{(a)}\right)\right] = \\
&= \sum_{\boldsymbol{\sigma}^{(1)}} ... \sum_{\boldsymbol{\sigma}^{(n)}} \int \left[\prod_{i<j} \frac{dJ_{ij}}{\sqrt{2\pi}} \exp\left(-\frac{J_{ij}^2}{2}\right) \exp\left(\beta \frac{J_{ij}}{\sqrt{N}} \sum_{a=1}^{n} \sigma_i^{(a)}\sigma_j^{(a)}\right)\right] = \\
&= \sum_{\boldsymbol{\sigma}^{(1)}} ... \sum_{\boldsymbol{\sigma}^{(n)}} \prod_{i<j} \frac{\sqrt{2\pi}}{\sqrt{2\pi}} \exp\left(\frac{\beta^2}{2N} \sum_{a=1}^{n}\sum_{b=1}^{n} \sigma_i^{(a)}\sigma_j^{(a)}\sigma_i^{(b)}\sigma_j^{(b)}\right)
\end{aligned}$$

where in the last step we used the fact that

$$\int dx \exp\left(-Ax^2 + Bx\right) = \sqrt{\frac{\pi}{A}} \exp\left(\frac{B^2}{4A}\right)$$

with $A = \frac{1}{2}$ and $B = \frac{\beta}{\sqrt{N}} \sum_{a=1}^{n} \sigma_i^{(a)}\sigma_j^{(a)}$. Continuing the rewriting of the n-th moment of the partition function, we obtain that

$$\begin{aligned}
\mathbb{E}[Z^n_{N,\beta,J}] &= \sum_{\boldsymbol{\sigma}^{(1)}} ... \sum_{\boldsymbol{\sigma}^{(n)}} \exp\left(\frac{\beta^2}{4N} \sum_{i<j} \sum_{a=1}^{n}\sum_{b=1}^{n} \sigma_i^{(a)}\sigma_j^{(a)}\sigma_i^{(b)}\sigma_j^{(b)}\right) \\
&= \sum_{\boldsymbol{\sigma}^{(1)}} ... \sum_{\boldsymbol{\sigma}^{(n)}} \exp\left(\frac{\beta^2}{4N} \sum_{i,j} \sum_{a=1}^{n}\sum_{b=1}^{n} \sigma_i^{(a)}\sigma_j^{(a)}\sigma_i^{(b)}\sigma_j^{(b)} - \frac{\beta^2}{4N} n^2 N\right) \\
&= \sum_{\boldsymbol{\sigma}^{(1)}} ... \sum_{\boldsymbol{\sigma}^{(n)}} \exp\left\{\frac{\beta^2}{2N} \sum_{a\neq b} \left(\sum_i \sigma_i^{(a)}\sigma_i^{(b)}\right)^2 + \frac{\beta^2}{4N} nN^2 - \frac{\beta^2}{4N} n^2 N\right\} \\
&= \exp\left(\frac{\beta^2}{4} n(N-n)\right) \sum_{\boldsymbol{\sigma}^{(1)}} ... \sum_{\boldsymbol{\sigma}^{(n)}} \prod_{a<b} \exp\left\{\frac{\beta^2}{2N} \left(\sum_i \sigma_i^{(a)}\sigma_i^{(b)}\right)^2\right\}
\end{aligned}$$

and we use the Gaussian result in the other direction with $B = \beta^2 \sum_i \sigma_i^{(a)}\sigma_i^{(b)}$, $A = \frac{\beta^2 N}{2}$ so

$$\mathbb{E}[Z^n_{N,\beta,J}] = \exp\left(\frac{\beta^2}{4} n(N-n)\right) \int \sum_{\boldsymbol{\sigma}^{(1)}} ... \sum_{\boldsymbol{\sigma}^{(n)}} \prod_{a<b} \frac{dQ_{ab}}{\sqrt{\frac{2\pi}{\beta^2}}} \exp\left(-\frac{\beta^2 N}{2} Q_{ab}^2\right) \exp\left(\beta^2 Q_{ab} \sum_i \sigma_i^{(a)}\sigma_i^{(b)}\right).$$

Continuing in this way we obtain

$$\begin{aligned}
\mathbb{E}[Z^n_{N,\beta,J}] &= e^{\frac{\beta^2}{4}n(N-n)} \int \prod_{a<b} \frac{dQ_{ab}}{\sqrt{\frac{2\pi}{\beta^2}}} e^{-\frac{\beta^2 N}{2}Q_{ab}^2} \sum_{\boldsymbol{\sigma}^{(1)}} \dots \sum_{\boldsymbol{\sigma}^{(n)}} \prod_{i=1}^{N} e^{\beta^2 \sum_{a<b} Q_{ab}\sigma_i^{(a)}\sigma_i^{(b)}} \\
&= e^{\frac{\beta^2}{4}n(N-n)} \int \prod_{a<b} \frac{dQ_{ab}}{\sqrt{\frac{2\pi}{\beta^2}}} e^{-\frac{\beta^2 N}{2}Q_{ab}^2} \prod_{i=1}^{N} \sum_{\boldsymbol{\sigma}^{(1)}} \dots \sum_{\boldsymbol{\sigma}^{(n)}} e^{\beta^2 \sum_{a<b} Q_{ab}\sigma_i^{(a)}\sigma_i^{(b)}} \\
&= e^{\frac{\beta^2}{4}n(N-n)} \int \prod_{a<b} \frac{dQ_{ab}}{\sqrt{\frac{2\pi}{\beta^2}}} e^{-\frac{\beta^2 N}{2}Q_{ab}^2} \left(\sum_{\boldsymbol{\sigma}^{(1)}} \dots \sum_{\boldsymbol{\sigma}^{(n)}} e^{\beta^2 \sum_{a<b} Q_{ab}\sigma_i^{(a)}\sigma_i^{(b)}} \right)^N \\
&= e^{\frac{\beta^2}{4}n(N-n)} \int \left(\prod_{a<b} \frac{dQ_{ab}}{\sqrt{\frac{2\pi}{\beta^2}}} \right) e^{-\frac{\beta^2 N}{2}\sum_{a<b} Q_{ab}^2} \exp\left(N \log \sum_{\boldsymbol{\sigma}^{(1)}} \dots \sum_{\boldsymbol{\sigma}^{(n)}} e^{\beta^2 \sum_{a<b} Q_{ab}\sigma_i^{(a)}\sigma_i^{(b)}} \right).
\end{aligned}$$

In conclusion, we derived that

$$\mathbb{E}[Z^n_{N,\beta,J}] = \int \prod_{a<b} \frac{dQ_{ab}}{\sqrt{\frac{2\pi}{\beta^2}}} \exp\{-N\mathcal{A}[Q]\}$$

where the argument of the exponential is equal to

$$\mathcal{A}[Q] = -\frac{\beta^2}{4}n(N-n) + \frac{\beta^2}{2}\sum_{a<b} Q_{ab}^2 - \log\left(\sum_{\boldsymbol{\sigma}^{(1)}\dots\boldsymbol{\sigma}^{(1)}} e^{\beta^2 \sum_{a<b} Q_{ab}\sigma_i^{(a)}\sigma_i^{(b)}} \right).$$

Now, we want to apply the RS-ansatz, i.e. $Q_{ab} = \mathbb{1}(a=b) + q\mathbb{1}(a\neq b)$. Thus

$$\mathcal{A}[Q] = -\frac{\beta^2}{4}n(N-n) + \frac{\beta^2}{2}\sum_{a<b} q^2 - \log\left(\sum_{\boldsymbol{\sigma}^{(1)}\dots\boldsymbol{\sigma}^{(1)}} e^{\beta^2 q \sum_{a<b} \sigma_i^{(a)}\sigma_i^{(b)}} \right).$$

If we assume commutativity of the limits for N,n then we obtain

$$A^Q_\beta = \lim_{n\to 0}\frac{1}{n}\lim_{N\to\infty}\frac{1}{N}(\mathbb{E}[Z^n_{N,\beta,J}] - 1) = -\lim_{n\to 0}\frac{1}{n}\mathcal{A}[Q^*]$$

where we used Laplace's method and $Q^* = \operatorname{argmin}\mathcal{A}[Q]$). By calculating the extremal point of $\mathcal{A}$ and doing the limit for n, we arrive at the formula

$$A^{Q,RS}_\beta = \frac{\beta^2}{4}(1-q^{RS})^2 + \log 2 + \log\int d\mu(z)\log\cosh\left(\beta z\sqrt{q^{RS}}\right)$$

with q^{RS} that satisfies the SC-equation

$$q^{RS} = \int d\mu(z)\tanh^2(\beta z\sqrt{q^{RS}}).$$

References

[1] Agliari, E. (2023). Modelli di Reti Neurali. Università degli Studi di Roma La Sapienza.

[2] Feng, J., Tirozzi, B. (1996). Capacity of the Hopfield model. Statistics Group, The Babraham Institute, Cambridge and Mathematisches Institut, Universität München, University of Rome La Sapienza.

[3] Alborè, N., Tubito, A. (2022). Biology and Machine Learning. Università degli Studi di Roma La Sapienza.

[4] Bellina, A. (2021). Modello di Hopfield con Diluizione Sinaptica Simmetrica Casuale. Università degli Studi di Roma La Sapienza.

[5] Agliari, E., Alemanno, F., Barra, A., Facchetti, A. (2020). Generalized Guerra's interpolation schemes for dense associative neural networks. Dipartimento di Matematica Guido Castelnuovo, Dipartimento di Matematica e Fisica Ennio De Giorgi, Università del Salento, C.N.R. Nanotec Lecce, I.N.F.N., Sezione di Lecce.

[6] Patti, A. (2022). Retrieving mismatched memory patterns in the Hopfield model of neural networks. Università degli Studi di Roma La Sapienza.

[7] Haykin, S. (1999). Neural Networks and Learning Machines, 3rd ed. Wiley.

[8] Negri, M., Lauditi, C., Perugini, G., Lucibello, C. and Malatesta, E. M. (2023). Storage and Learning Phase Transitions in the Random-Features Hopfield Model. University of Rome La Sapienza, Bocconi University.