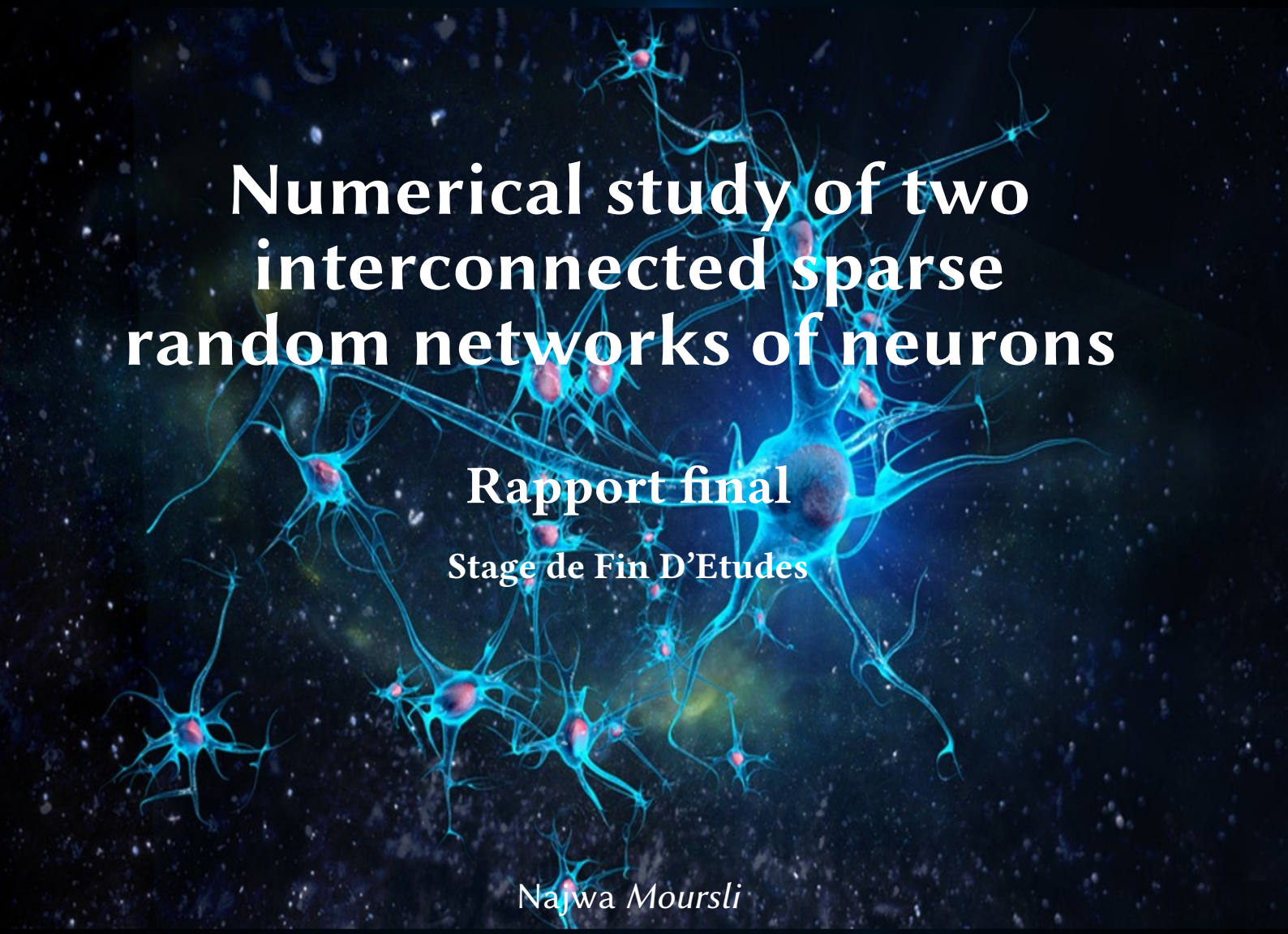


Numerical study of two interconnected sparse random networks of neurons

Rapport final

Stage de Fin D'Etudes

Najwa *Moursli*

Polytech Sorbonne
Spécialité
Mathématiques Appliquées et Informatique Numérique
Année 5
2020 – 2021

POLYTECH SORBONNE

cnrs

SCIENCES SORBONNE UNIVERSITÉ

1. Acknowledgements

Je remercie

Pr. Fanny *Villers* pour son rôle de coordinatrice, qui a su m'accompagné tout au long de ce stage,

Pr. Alain *Destexhe*, mon tuteur de stage, pour son encadrement bienveillant et ses conseils avisés,

Ph.D. Damien *Depannemaecker* et Ph.D. Malory *Carlu*, mes co-encadrants pour leur soutien et aide précieuse.

Contents

List of Figures

List of Tables

2. Introduction

The subject of computational neurosciences is to decipher the working principles of the brain. This goal is particularly difficult, since the brain constitutes one of the most complex systems we know: Its dynamics arises from an interaction between a tremendous number of individual components, the neurons.

Spiking neural networks (*SNN*) are inspired by information processing in biology, where sparse plural and asynchronous binary signals are communicated and processed in a massively parallel fashion [1]. Yet biological neurons use discrete spikes to compute and transmit information, and the spike times, in addition to the spikes rates, matter. *Spiking neural networks* (*SNN*) are thus more biologically realistic than artificial neural network (*ANN*) used in machine learning and arguably the only viable option if one wants to understand how the brain computes. However, training deep *spiking neural network* is still challenging.

Local *cortical columns* in cerebral cortex can be modeled as a sparse plural random neural networks of excitatory (*regular-spikings* or *RS*) and inhibitory (*fast-spikings* or *FS*) neurons [2]. Such neural networks can display different states such as asynchronous irregular (AI) or oscillatory (synchronous) states. However, it is not clear how such dynamics can be extrapolated to large-scale neural networks (*NN*) where many *cortical columns* interact through excitatory connections.

One route towards understanding this system is to formulate models which describe the dynamics of the brain in terms of mathematical equations. Bottom-up approaches aim to model the individual neurons and investigate the emerging collective dynamics. However, as the brain displays dynamics on various spatial and temporal scales, also the emerging dynamics in the models is difficult to understand. To better assess the collective dynamics, theoretical neuroscience has developed analytical tools that reduce the complexity of neural networks.

First and foremost, at the low mesoscopic level, we will simulate a simple *cortical column* composed of two populations (*fast-spikings* and *regular-spikings*) in order to determine how this *cortical column* system behaves and what are the critical parameters involved. Subsequently, we will consider the corresponding mean-field model where the *cortical column*. This system if simpler and much faster to simulate, but it remains to be seen if the mean-field version of the *cortical column* network displays dynamics as the *spiking neural network*.

Unlike in vivo and in vitro experimental setups, in silico models offer the possibility of studying a wide range of neuronal and stimulation parameters and might help to interpret experimental results.

Therefore, inspired by the Virtual Brain approach [I], the examination of scaling up possibility will be assessed with a two interconnected *cortical columns* model, connecting therefore four populations (two *fast-spiking* and two *regular-spiking* neural groups) which could help ascertaining the bifurcation brain states map for a better understanding of the interactions of time-varying parameters with the dynamics of large neural populations.

2.1 Definitions and acronyms

Ahead of immersing ourselves in the main topic, it is important to introduce the definition and acronym of the terms used. This list of definitions and acronyms can be found at the end of this report (cf. Appendix pages 45).

Project available: https://github.com/NajwaMoursli/Interconnected-Mean-Fields

[I] https://www.thevirtualbrain.org

3. Research Laboratory and Context

3.1 Presentation of the laboratory

As part of my engineer formation at Polytech-Sorbonne, I was brought to fulfill an end-of-studies engineering internship which happened to be within an interdisciplinary research laboratory, U.N.I.C.-C.N.R.S., Unit of Neurosciences Information and Complexity, [I], based in Gif-Sur-Yvettes. This laboratory is jointly financed by the Human Brain Project (HBP) [II] and the C.N.R.S.

The research conducted in this laboratory stands at the interface between several disciplines, such as biophysics, physics and neuroscience. The themes investigated range from the microscopic (single neurons) to the macroscopic (networks or populations of neurons) aspects of the central nervous system function. They use theoretical methods and computer-based simulation techniques to explore the complex behavior of single neurons and understand their basic integrative properties. At the network level, they try to understand the collective behavior of neuronal populations, which in many cases cannot be simply deduced from single-cell behavior.

Alain *Destexhe* is a Director of Research of the U.N.I.C.-C.N.R.S. He is an expert in Physical Chemistry and Theoretical Biology [3]. He was my internship tutor and was in charge of making the research project outline and managing the research team. Damien *Depannemaecker* is a post-doc researcher, one of my co-supervisors, specialized in Experimental Neurosciences [4], and the *AdEx* referee within the team.Mallory *Carlu* is a post-doc researcher and my other co-supervisor, specialized in Applied Physics more precisely in Complex Systems and Mathematical Biology [5], and the Mean-Field referee within the team.

3.2 Context of Research

Computational neuroscience is such a timely topic because, with the amount of data being recorded continually increasing, along with its complexity, we really need theory and analysis tools to handle it all. In recent times, we heard how a neurotech startup, Neuralink [III], plans to implement brain-machine interfaces converting neuronal information to commands onto the brain [6]. The biggest question that arises is how will the signals in the brain be processed. Using *recurrent neural networks* to build models has become mainstream. However, they do not imitate the mechanism of the brain's neurons as the *spiking neural network* and their reduced models which aim to ease the application of machine learning in neuroscience.

A lot of the classical theories of neuroscience were based on attractor solutions to neural networks. That idea has been critical for thinking about systems that do integration; Goldman, Raymond, and Aksay [7]'s SCGB work is one example [7]. But attractor solutions represent just one aspect of what neural networks can do due to their complex dynamics. We would rather take advantage of these to perform a task. The field tends to go towards using data to fit neural networks, which then serve as a computational tool to extract meaning (e.g. Mante, Sussillo, Shenoy, *et al.* [8] looked at a complex decision-making task).

This approach is a key conceptual advance in the field that is gaining interest and traction. However, in this exciting new domain of high-level neural network models, no one cares about the exact properties of individual cells. It is necessary to be able to link micro and macroscopic scales and that is exactly what *spiking neural networks* and especially mean-field models, derived from *spiking neural networks* aim to do.

Since, Computer science has reached the computational power necessary for the large scale simulation of distributed systems as complex as neural networks, researchers have developed models of large-scale brain networks that are informed by the underlying anatomical connectivity contribute to our understanding of the mapping between the structure of the brain and its dynamical function. Connectome-based modelling, as the Virtual Brain [IV], an open-source simulation platform composed of thousands of interconnected mean-field developed by the Human Brain Project (HBP) [V], is a promising approach to a more comprehensive understanding of brain function.

[I] cf. http://cns.iaf.cnrs-gif.fr/MainF.html
[II] https://www.humanbrainproject.eu/en/
[III] https://neuralink.com/
[IV] cf. https://thevirtualbrain.org
[V] cf. https://www.humanbrainproject.eu

4. Internship Research Topic

4.1 Neuroscience Background

4.1.1 Action Potential : the reference measure for SNN

Neurons are the elementary unit of the brain, forming the substrate for information processing. The basic structures of each neuron are the soma (cell body), the dendrites by which they receive input from other neurons and the axon through which they send output to other neurons.The membrane potential is the voltage difference between the intracellular and extracellular medium. In general, the value of the membrane potential changes due to currents which flow through membrane-spanning ion channels. The channels are specialized to certain ions among which the most prominent ones are sodium Na^+ and potassium K^+.

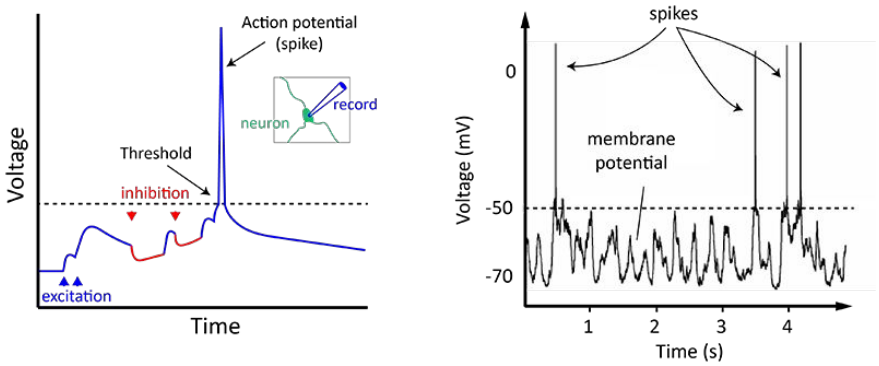


Figure 4.1: Mechanism of the action potential in a neuron

A sufficiently positively charged neuron will eventually create a stimulating inward directed current by depolarizing. During the genesis of this spike, post-synaptic *depolarization* add up and may generate a new action potential if their sum reaches or exceeds a minimum value called the *depolarization* threshold or the the refractory period which corresponds to the time-span directly after the spike when the neuron is not excitable and thus not able to generate another spike.

Typically, a neuron will generate a spike at a frequency called the firing rate, when the threshold, the value of the membrane potential required to activate voltage-dependent Na^+ channels is reached. The *depolarization* is followed by a *repolarization*, thanks to the output of an equivalent quantity of K^+ ions, which causes the return to the steady state. Then, *hyperpolarization* is triggered by the output of more K^+ than Na^+, an outward directed current. There is finally a return to the rest potential for (without input from other neurons, the membrane potential takes a value of about -70 mV) which results from an equilibrium between electrical and chemical forces acting on the ions [9]. Dayan and Abbott [9] states that one neuron only releases one neurotransmitter type so that neurons can be classified into excitatory or *regular-spikings* and inhibitory or *fast-spikings* cells.

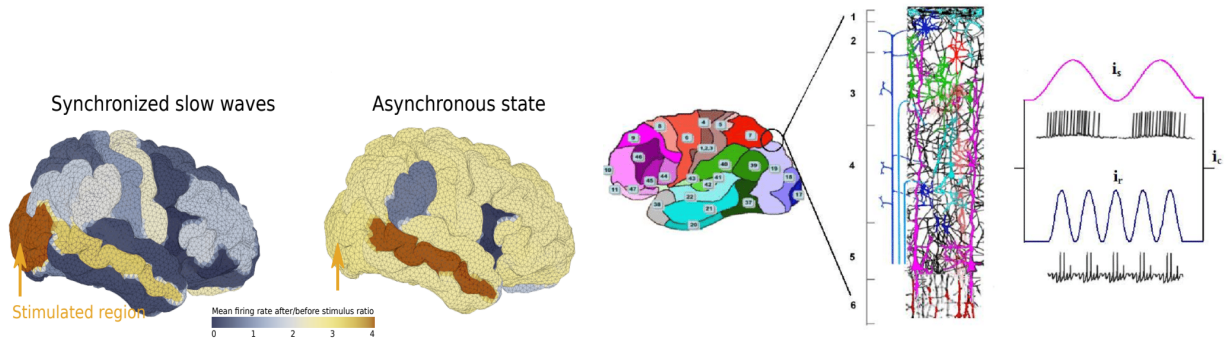
The spike propagates along the axon and eventually arrives at the synapses, contact points between the pre-synaptic axon and the post-synaptic dendrites with a synaptic cleft in between. On the pre-synaptic side, the voltage transient of the spike causes the release of neurotransmitters into the cleft which in turn bind to receptors on the post-synaptic. The corresponding change in the membrane potential of the post-synaptic neuron is termed post-synaptic potential (PSP). The time span between the emission of a spike and the evoked PSP is called the transmission delay.

4.1.2 Cortical columns and brain states

The cerebral cortex has been geometrically designed so that the locations of cortical structures and the connections between them allow for the greatest degree of association, both in integrating different areas of the cerebral cortex and in producing higher level abstractions. As a matter of fact, cortical division of function is a natural outcome of certain areas of cerebral cortex being mapped to certain extra cortical functions, i.e. sensory and motor maps. As a result of the spreading activation of call trees, the areas adjacent to these maps assume the roles of progressively higher level integration and association units [10].

On a lower hierarchical level, the *cortical column* is the basic functional unit of the cerebral cortex (neuron is also considered as the basic unit of the cerebral cortex, this topic is under discussion). The *cortical column* is oriented perpendicular to the cortical surface. All neurons inside the *cortical columns* are tightly connected, although neurons connections extend to adjacent *cortical columns* and *cortical columns* far across the cerebral cortex and into sub-cortical areas [10].

The cerebral cortex displays asynchronous irregular (AI) states during wakefulness that could be described as de-synchronized states without specific patterns and synchronized brain states consisting of the alternation between periods of tonic firing spikes (UP states) and periods of quiescence (DOWN states) which are taking place during slow-wave-sleep [11].



(a) Mapping of synchronous vs asynchronous states for the whole-brain activity [12]

(b) From cortex to column to simple circuit [13]

Figure 4.2: The cortex structure and activity

4.2 State of Art

The models under study describe the relationship between neuronal membrane currents at the input stage, and membrane voltage at the output stage [14]. *regular-spiking-fast-spiking* neurons model constitutes the simplest spiking model that accounts for the fact that *FS* neurons are more excitable than *RS* ones. All the parameters mentionned are referred and assessed in the Table A.1.

4.2.1 Leaky Integrate-and-Fire model

To build networks, the simplest model is the leaky integrate-and-fire (LIF) model introduced in the early 1900's by L. Lapicque [15], where the neuron is described by a passive membrane with a threshold. There is a close analogy to the electrical circuit R-C, R_m and CR_m are the membrane resistance and capacitance. When the threshold is reached V_{th}, the neuron fires and its membrane potential is set to a reset value V_r. When the membrane potential reaches a given threshold, its value is reset. A "leak current", I_l, is then added to account for membrane currents which flow out of the neuron, and reproduce the value of the resting membrane potential.

According to Schücker, Honerkamp, and Helias [16], the LIF neuron model combines two important aspects of neuronal dynamics, the sub-threshold integration of inputs by the membrane potential as well as the interaction with other neurons via spikes (which are fundammental for the mean-field model we will see afterwards). However, its use is limited since it can not generate neither Up and Down nor asynchronous irregular states as shown by Fuortes and Mantegazzini [17]. In order to solve this issue, spike-adaptation frequency or slow adaptation due to *Regular-spikings* (*RS*) cells, another variable w, the synaptic adaptation, has to be take into account as in generalized integrate-and-fire models thanks to L. Lapicque [15].

4.2.2 Adaptive exponential integrate-and-fire model

AdEx is simply a two-dimensional improved integrate-and fire model. Initially, the one dimensional model was proposed by Brette and Gerstner [18] then modified to include an exponential non-linearity around spike threshold, based on AdExIF model of Fourcaud-Trocmé, Hansel, Vreeswijk, *et al.* [19]. These two models were combined later by Izhikevich [20] to get a 2 dimensional model leading to the set of equations (cf. eq. 5.1 and 5.2) with the parameters in Table A.1.The *AdEx* model is a more realistic model than the leaky integrate and fire model, because of its exponential approach to threshold V_t added to the slow adaptation parameter. The *AdEx* models of Izhikevich [20] a lots of intrinsic neuronal properties, such as *regular-spikings* and *fast-spikings* excitability difference, which has potentially important consequences at the large-scale level [21].

Following this attempt of understanding intrinsic properties, Destexhe [22] focused on the study of the adaptation parameters a and b, the adaptation conductance and its increment, in order to asses the polarisation stimuli phenomenon different in the various cells types that largely determines the firing patterns of the neuron which can be related to the dynamics of ion channels, as reported by Kristensen and MacNearney [23] (a phenomenon previously described in subsection 4.1.1).In the same paper, excitatory cortical neurons were modeled as *regular-spikings* with spike-frequency adaptation. Therefore, it turned out, increasing the parameter a leads to bursting activity while increasing b generates a model with bursting activity as well but in response to both *depolarization* and *hyperpolarization* stimuli. For b = 0, the model generated responses with a negligible level of adaptation, similar to the *fast-spikings* cells encountered in cerebral cortex, and which corresponds mostly to cortical inhibitory neurons.

On the practical side, *AdEx* model is a good candidate for large-scale simulations of realistic cortical networks. On the theoretical side, the analysis of networks of *AdEx* neurons could also help us understand how intrinsic properties defined electrophysiologically in vitro affect the neuron behavior in vivo.

4.2.3 Mean-field model

The Mean field or *MF* model studies the behavior of high-dimensional random (stochastic) models such as *AdEx* through a simpler model that approximates the original by averaging over degrees of freedom, therefore reduction of dimension with only 3 stochastic differential equations (*SDE*) under consideration. In mean-field theory, the effect of all the other individuals on any given individual is approximated by a single averaged effect, thus reducing a many-body problem to a one-body problem.

These models consist in deriving population-level models based on the properties of single neurons and their interactions by linking mesoscopic and cellular scales. Mesoscopic dynamics describe the *MF* activity of populations of neurons organized as *cortical columns*. Common assumptions in mean-field modeling are explicit structural or temporal features of neural networks which are irrelevant for the analysis of complex mesoscopic dynamics and the emergent collective behavior (e.g. small brain regions up to the whole brain) is only weakly sensitive to the details of individual neuron behaviour as stated by Breakspear and Jirsa [24].Therefore, modeling population of neurons is representative enough and less time/performance consuming than cellular level as discussed by Volo, Romagnoni, Capone, *et al.* [25]. Basic *MF* models capture changes of the mean firing rate [26], whereas more complex *MF* models account for parameter dispersion in the neurons [27], [28], for instance Zerlaut, Chemla, Chavane, *et al.* [21] investigated a mean-field model of networks with different electrophysiological properties, described using the *AdEx* model with conductance-based synapses proposed by Górski, Depannemaecker, and Destexhe [29].

For the theoretical analysis of asynchronous dynamics in sparsely connected random networks, the Markovian formalism proposed in Boustani and Destexhe [30] was shown to be a relatively accurate description of the response simulated in numerical networks. However, the proposed semi-analytical given by Soula and Chow [31] approach offers a convenient description for theoretical models where an exact analytical treatment would not be achievable. Therefore, Poisson noise (or on a macroscopic level *Ornstein–Uhlenbeck* noise) is used as stimulus because of the complex pattern of firing rate population spikes which is a fundamental hypothesis of the *AdEx* based on the chaos theory described by Izhikevich [20].

In 2005, Massimini, Ferrarelli, Huber, *et al.* [32] at University of Milan, carried out experiments on human subjects. They measured thanks to electroencephalograms, how the brain reacts to certain stimuli, in order to understand how different parts of the brain interact (cf. the different brain states assessed in 4.1.2). Based on these, Goldman, Kusch, Yalcinkaya, *et al.* [12], by parametrizing the thousands of mean-field interconnected in the Virtual Brain using the EBRAINS platform, they were able to simulate the two different responses to observed in the experiments.

4.3 Project Main Goals

The interest of this modeling endeavor is not so much in the microscopic/macroscopic observations, but rather in their relations. Indeed, we are interested in which microscopic features allows, produces or sustains macroscopic phenomena. This is a central question raised in and by multi scale modeling.

4.3.1 Building of a model

In the presence of data recorded from mesoscopic to macroscopic scale experiments someone we would like to build a model of neural network based on these recordings. Before diving into it, there are several steps and questions that will lead our reflection, which are gathered around the main question **What is the purpose of building this model ?**:

Understanding and identifying important features of a system ⟶ How does the external environment such shapes the global dynamics?

Testing assumptions on structural aspects of the dynamics ⟶ Do the mean-field models represent well the dynamics displayed by the *spiking neural networks* ? Where does the validity scope of the Mean-Field model range?

Exploring new possible behaviors of the system ⟶ In our study, how parameters changes could display different brain states and for which values do these states switch to another?

Embedding it in larger structures For observing differences or similarity of behaviors in different dimension ⟶ Comparing the interconnected mean-field models which enables to study macroscopic dynamics and the interconnected *AdEx* that enables to study microscopic dynamics.

4.3.2 Use and Study of the model

Understanding the brain is always a tricky question, sometimes we cannot predict how brains react in a few scenarios, in spite of them indulging in routine activities. They store a lot of information inside neurons based on the actions they keep triggering. So the question here is, how do we need to interpret the information? There are three computational models to understand brains which explain the three questions, **"What, How, and Why"**. These models are named as Descriptive, Mechanistic, and Interpretive models respectively.

1. Descriptive Models ⟶ **"What are the neuronal responses to external stimuli?"** They review large amounts of experimental data, thereby characterizing what neurons and neural circuits do :
   - What qualitative data could we describe by Neural Encoding ?
   - What information could we extract from neurons by Neural Decoding techniques ?
2. Mechanistic Models ⟶ "How nervous systems operate by known anatomy, physiology, and circuitry". Such models often form a bridge among descriptive models at different levels :
   - How can we simulate the behaviour of a single neuron on a computer?
   - How do we simulate a biological neural network?
3. Interpretive Models ⟶ "Why nervous systems operate as they do ?" These use computational and information-theoretic principles to explore the behavioral and cognitive significance of various aspects of nervous system functionality:
   - Why do brains operate the way they do?
   - What are the computational principles underlying them?

Therefore, we would like to determine what kind of information do we want to get and which model do we need. This leads to the inverse consideration, does the model I am using is descriptive, mechanistic, interpretive, a combination of two/three of them or none and is the model type I am using appropriate to account the kind of data in a given situation correctly?

5. Methods

5.1 Analytical Model

5.1.1 AdEx model

The dynamics of each neurons types is based on the model described by the following equations Brette and Gerstner [18] and Izhikevich [20]. The parameters description are available in the table A.1. The first equation describes the dynamics of the membrane potential and includes an activation term with an exponential voltage dependence. Voltage is coupled to a second equation which describes adaptation. Both variables are reset if an action potential has been triggered:

$$C_m \frac{dv}{dt} = -g_l \times (v - E_l) + gl \times Dt \times e^{\frac{w - v_t}{D_t}} - w + I_{syn}) \quad (5.1a)$$

$$\tau_w \frac{dw}{dt} = a \times (v - E_L) - w \quad (5.1b)$$

$$\frac{dG_{syn_s}}{dt} = -\frac{G_{syn_s}}{T_{syn}} \quad (5.2a)$$

$$I_{syn} = -G_{syn_e} * (v - E_e) - G_{syn_i} * (v - E_i) \quad (5.2b)$$

$$G_{syn_S}(t) = Q_s \sum_{s.pre} \mathcal{H}(t - t^s_{sp}(k)) \times e^{\frac{t - t^s_{sp}(k)}{\tau_s}} \quad (5.2c)$$

Here are the Synaptic equations, with s={e,i}. The synaptic current I_{syn} received by neuron i is the result of the spiking activity of all pre-synaptic neurons j ∈ pre(i). This current can be decomposed in the result received from excitatory and inhibitory pre-synaptic spikes. Notice that we consider voltage dependent conductance. Finally, we model G_{syn_E} and G_{syn_I} as decaying exponential function that takes kicks of amount Q_e and Q_i (synaptic weights) at each pre-synaptic spike, where $\mathcal{H}$ is the heaviside function.

5.1.1.1 Network Quantification

For the *AdEx*, we benefit from over measures to quantify the synchrony and regularity of the state we are in [22]:

1. Regularity : To quantify the degree of temporal regularity, we used the coefficient of variation (CV) of the interspike intervals (ISI), averaged over all cells the network:

$$CV_{ISI} = \frac{\langle \sigma_i^{ISI} \rangle}{\langle ISI_i \rangle} \quad (5.3)$$

   The ⟨.⟩ indicate an average over all neurons, while $\langle ISI_i \rangle$ and σ_i^{ISI} are respectively the mean and standard deviation of the interspike intervals of neuron i. The CV_{ISI} is expected to be ≥ 1, for temporally irregular systems (1 for a Poisson process). If the CV_{ISI} is ≤ 1, we encounter an inhomogenous Poisson Process.

2. Synchrony : The degree of synchrony is quantified using the averaged pairwise cross-correlation (CC) between neurons in the network:

$$1 \geq CC = \frac{\langle Cov(S_i, S_j) \rangle}{\sigma(S_i)\sigma(S_j)} \geq -1 \quad (5.4)$$

   The CC takes high values only for synchronous states. A given network state can reasonably be considered as "asynchronous" if CC is low enough (typically ≤ 0.1) [30].

5.1.2 Mean-Field model

In this setting, a mean-field is essentially a description of the message passed to a given neuronal population. Here, each factor may be thought of as predicting the other (via the transfer functions). This prediction is subtracted from the current expectation (μ_V) to give an error term (h). The assumption here is that the time constants of the neural populations representing this error are very short relative to those of the expectations. The error term induces updates in the expectation such that it conforms to the prediction. The dynamic mean field model was developed according to the following two steps: In the first step, a microscopic mean field model is derived from the stochastic model by taking the expectation (noise average), with respect to each variable. This expectation cannot be replaced with the population or time average. In the second step, a macroscopic mean field model is derived from the microscopic model by taking the average over the population.

5.1.2.1 The transfer function

The transfer function characterizes how a modulation of the neuron's input rate is transmitted to a modulation of its output rate. A neuron in the brain receives signals from a large number of presynaptic neurons, effectively causing a noisy input. Extending to two interconnected *cortical columns*, each described by a mean-field model, we define networks of mean-field models. The equations of such a network, expanding the two-population mean-field described by the set of equations (5.5) to several mean-field, we get the other set of equations (5.6).

$$T_{syn}\frac{d\nu_e}{dt} = F_e(\nu_e,\nu_i) - \nu_e \quad (5.5a) \qquad T_{syn}\frac{d\nu_e(k)}{dt} = F_e(\nu_e^{input}(k),\nu_i(k)) - \nu_e(k) \quad (5.6a)$$

$$T_{syn}\frac{d\nu_i}{dt} = F_i(\nu_e,\nu_i) - \nu_i \quad (5.5b) \qquad T_{syn}\frac{d\nu_i(k)}{dt} = F_i(\nu_e^{input}(k),\nu_i(k)) - \nu_i(k) \quad (5.6b)$$

$$\frac{dw}{dt} = \frac{-w}{\tau_w} * b * \nu_e + a(\mu_V(\nu_e,\nu_i,w) - E_l) \quad (5.5c) \qquad \frac{dw(k)}{dt} = \frac{-w(k)}{\tau_w} * b * \nu_e(k) + a(\mu_V(\nu_e(k),\nu_i(k),w(k)) - E_l) \quad (5.6c)$$

where $\nu_e(k)$ and $\nu_i(k)$ are the excitatory and inhibitory population firing rates at site k, respectively, w(k) the level of adaptation of the population, and $\nu_e^{input}(k)$ is the excitatory synaptic input. The latter is given by:

$$\nu_e^{input}(k) = \nu_e^{drive}(k) + \sum_j C_{jk}\nu_e\left(j, \frac{t - \|j-k\|}{v_a}\right) \quad (5.7)$$

where the sum runs over all nodes j sending excitatory connections to node k, and C_{jk} is the strength of the connection from j to k (and is equal to 1 for j = k). Note that $\nu_e\left(j, \frac{t-\|j-k\|}{v_a}\right)$ is the activity of the excitatory population at node k at time $t - \|j-k\|$ to account for the delay of axonal propagation. Here, $\|j-k\|$ is the distance between nodes j and k and v_a is the axonal propagation speed.

Eventually, generalizing these equations using the Einstein indexes and displaying the interaction terms where α = {e,i} is the population index (excitatory or inhibitory), ν_{alpha} the population firing rate, $c_{\lambda\eta}$ the covariance between populations and $\delta_{\eta\lambda}$, the Dirac function for two populations with λ and η, the Einstein's indexes :

$$\begin{cases} T\frac{\partial\nu_\alpha}{\partial t} &= (F_\alpha - \nu_\alpha) + \frac{1}{2}c_{\lambda\eta} \times \frac{\partial^2 F_\alpha}{\partial\nu_\lambda\partial\nu_\eta} \\ T\frac{\partial c_{\eta\lambda}}{\partial t} &= \delta_{\eta\lambda}\frac{F_\lambda(\frac{1}{T}-F_\eta)}{N_\lambda} + (F_\eta - \nu_\eta)(F_\lambda - \nu_\lambda) + c_{\eta\alpha}\frac{\partial F_\lambda}{\partial\nu_\alpha} + c_{\lambda\alpha}\frac{\partial F_\eta}{\partial\nu_\alpha} - 2 * c_{\lambda\eta} \\ \frac{dw}{dt} &= \frac{-w}{\tau_w * b * \nu_e} + a(\mu_\nu(\nu_e,\nu_i,w) - E_l) \end{cases} \quad (5.8)$$

The transfer function F of a mean-field model is defined here as the function that maps the value of the stationary *RS* and *FS* neurons' presynaptic release frequencies and adaptation (resp. ν_e and ν_i) to the output stationary firing rate response ν_{out}, regardless of the neurons location, computed by the semi-analytical method [31]. Based on previous observations made by Fourcaud-Trocmé, Hansel, Vreeswijk, *et al.* [19] and Brunel and Sergi [26], two parameters seem critical in the transfer function: the time constant τ and the corrective term h, which takes into account colored noise in the synaptic input [30].

$$\nu_{out} = F_{\nu_\alpha}(\nu_e,\nu_i) = \frac{1}{2\tau_V} \times Erfc\left(1 + \frac{\mu_V - V_t^{eff}}{\sqrt{2}\sigma_V} + \delta h\right) \quad (5.9)$$

Erfc is the Complementary Error Function. In probability and statistics, *Erf*, (Error Function) ≡ Erf(x) = 1 - Erfc(x), integrates the normal distribution. It gives the probability that a normally distributed random variable ɣ (with mean 0 and variance $\frac{1}{\sqrt{2}}$), falls into the range [-x, x] [33].

$$Erfc(x) = \frac{2}{\sqrt{\pi}}\int_{x=t}^{t=\infty} e^{-t^2}dt \quad (5.10)$$

In the equation 5.11, (μ_V , σ_V, τ_V) are the mean, standard deviation and auto-correlation time constant (taken into account if $\tau_{V_e} \neq \tau_{V_i}$) of the membrane potential fluctuations also referred as the subthreshold moments. Here we make the assumption that these are not affected by the dynamics of the currents coming into play at the spiking time (e.g. sodium channels dynamics (cf. subsection 4.1.1) or the exponential term of the *AdEx* model (cf. equation 5.1). We thus consider, for all neurons, only the leakage term and the synaptic input in order to estimate subthreshold moments defined [21] as followed:

$$\mu_V(\nu_e,\nu_i) = \frac{\mu_{G_e} * E_e + \mu_{G_i} * E_i + g_l * E_l - w}{\mu_G} \quad (5.11a)$$

$$\mu_G = \mu_{G_e} + \mu_{G_i} + gl \quad (5.11b)$$

$$\sigma_V(\nu_e,\nu_i) = \sqrt{\sum_s K_s\nu_s\frac{(U_s\tau_s)^2}{2(\tau_m^{eff}+\tau_s)}} \quad (5.11c)$$

with $\tau_m^{eff} = \frac{c_m}{\mu_G}$

$$\tau_V(\nu_e,\nu_i) = \sum_s K_s\nu_s(U_s\tau_s)^2 \times \frac{2(\tau_m^{eff}+\tau_s)}{\sum_s K_s\nu_s(U_s\tau_s)^2} \quad (5.11d)$$

with s={e,i}, the following averages $\mu_{Ge,Gi}$ and standard deviations $\sigma_{Ge,Gi}$ of the conductances given by 5.2 under the assumption that the input spike trains follow the Poisson statistics (as is indeed the case in asynchronous irregular regimes here considered) [21], [22]

$$\mu_{Gs}(\nu_e,\nu_i) = \nu_s K_s\tau_s Q_s \quad (5.12a)$$

$$\sigma_{Gs}(\nu_e,\nu_i) = \frac{\sqrt{\nu_s K_s\tau_s}}{2}Q_s \quad (5.12b)$$

with $U_s = \frac{Q_s}{\mu_G}(E_s - \mu_V)$

5.1.2.2 The Markov Formalism

Neural networks are in particular brain state (cf. subsection 4.1.2) thus each state is modeled by a probability to be in this state. E corresponds to the state where the neurons spike, therefore the transfer function, ν when it spikes which corresponds to:

$$E = \frac{N_\alpha}{T \times N} = \frac{\nu \times T \times N}{T \times N} = \nu \tag{5.13}$$

with N : the population size, N_α or $\nu \times N \times T$: the number of spikes emitted by a population α and T : the time window size.

We would like to know the transition probability $\mathbb{P}_t(E|E')$, the probability for neurons to be in state E while previously being in E' which represents as well the spiking group which fires with the same probability $\mathbb{P}_t(E)$.The time window T shows the Markov properties of the network if we use ν as the transfer function to pass from state E to E' due to the assumption of the system to be time invariant, so this probability depends only on the time constant T.

As a result, we can thus assume that the population-conditional probabilities are independent from each other beyond the timescale of T [30], with α a population among the K homogeneous ones (here four for the interconnected models and two for one *cortical column*) and γ another population among the K remaining ones .

$$\mathbb{P}_T(E_\alpha|E'_\alpha) = \mathbb{P}_T(E_{1,\dots,K}|E'_\alpha) \tag{5.14}$$

Each neuron j connects excitedly (a spike occurs, the transfer function goes from $\nu to \nu'$) to other neuron i (and to itself) with a constant probability $\mathbb{P}_T(E_\alpha)$, the probability of **one** neuron spiking during T, generating a fixed set of connections through a Bernoulli process:

$$\mathbb{P}_T(E_\alpha|E'_\gamma) = \binom{N_\alpha}{\nu_\alpha N_\alpha T} \times \mathbb{P}_\alpha(E'_\gamma)^{(\nu_\alpha N_\alpha T)} \times (1 - \mathbb{P}_\alpha(E'_\gamma))^{N_\alpha(1-\nu_\alpha N_\alpha T)} \tag{5.15}$$

Obviously, $(1 - \mathbb{P}_\alpha(E'_\gamma))$ is the probability for **one neuron to not fire while being in** E'_γ **during the period T**.

We need to specify the transition functions $W(E_\alpha|E'_\gamma)$. These functions depend directly on the neuron properties. If we assume that we know the stationary transfer function of neurons in population α, ν_α, then the probability $\mathbb{P}_\alpha$ that a neuron in this population fires during time T given the previous state of the network E'_γ is a direct consequence of 5.15 and the fundamental hypothesis for the *master equation* [30] :

$$\mathbb{P}_\alpha(E'_\gamma) = \nu_\alpha(E'_\gamma) \times T \leq 1 \tag{5.16}$$

Therefore, the transfer function ν_α in the eq. 5.16 reckons the the spike on a single neuron and on the entire population. If a neuron spikes in one state E $\Leftrightarrow$ one population α, the mean firing rate during T is evaluated by ν_α. **This condition requires one neuron to spike only once during the period of time T**.

We can benefit from the Fokker-Plank equation to compute different transfer functions for our neurons thanks to the transition operator $W(\nu|\nu')$ which provides the rate of transition from state E'_γ=ν' to states E_α=ν giving the master equation its intuitive interpretation. It can be defined using the conditional probability density in equation 5.14:

$$W(\nu'|\nu) = \lim_{T\to 0} \frac{\mathbb{P}_T(\nu|\nu')}{T} \tag{5.17}$$

$$= \lim_{T\to 0} \frac{\prod_{\alpha=1,\dots,K} \mathbb{P}_T(E_\alpha|E'_\gamma)}{T} \tag{5.18}$$

Indeed, in this formalism, it encompasses internal properties of the neuronal models, together with the type of synaptic interactions under consideration, to yield a population scale description of the continuous master equation 5.19.

$$\partial_t \mathbb{P}_t(\nu) = \int_0^{\frac{1}{T}} \partial\nu' \times \mathbb{P}(\nu') \times W(\nu|\nu') - \mathbb{P}(\nu) \times W(\nu'|\nu) \tag{5.19}$$

$\mathbb{P}(\nu') \times W(\nu|\nu')$ models the neurons flow entering in states E_α and $\mathbb{P}(\nu) \times W(\nu'|\nu)$, neurons flow leaving states E_α.

5.2 AdEx Structure

5.2.1 Creation of Neurons, Synapses and Soma

Here displayed the lists of the fundamental variables which characterized neurons, synapses and neural network [30] and [34].

- **eqs** which refers to eq. 5.1 and 5.2 used in **NeuronGroup** and **Synapses** to define state variables and continuous-updates on these variables, through differential equations. When defining it, we check if all the system units of each parameters assigned is coherent. For each population, we set initial values for every terms of the equation.
- **threshold** : above this value the neuron spike
- **reset**: below neuron returns to the steady state
- **refractory period** : during this period the neuron does not spike [35].
- **integration method** : Here the stochastic Heun method is picked for solving stochastic differential equations.
- **synaptic weights** : Neurotransmitters are released into a synapse in packaged vesicles called quanta. One quantum generates the smallest amount of stimulation that one neuron can send to another neuron, Q_i for inhibitors and Q_e excitators.

In Brian, you only create groups of neurons and synapses, using the class **NeuronGroup** and **Synapses**. For the former, the first two arguments when you create one of these objects are the number of neurons and **eqs**. The variable v, the potential, is assumed by default that it is a postsynaptic variable, defined in the target. In this case, the constant w is added to variable v which is the synapse-specific weight parameter.

We construct an *AdEx* composed of four populations thus two *RS-FS cortical columns*. For the configuration, we decide that corresponding cells groups structure and parameters values are identical.

```
#start Brian scope:
start_scope()
#set dt value for integration (ms):
DT=0.1
seed(1)
defaultclock.dt = DT*ms
#total duration of the simulation (ms):
TotTime=10000
duration = TotTime*ms
#########################################################
#set the number of neuron of each population
N1 = 2000
N2 = 8000

eqs='''
dv/dt = (-GsynE*(v-Ee)-GsynI*(v-Ei)-gl*(v-El)+ gl*Dt*exp((v-Vt)/Dt)-w + Is)/Cm : volt (unless
    refractory)
dw/dt = (a*(v-El)-w)/tau_w:ampere
dGsynI/dt = -GsynI/Tsyn : siemens
dGsynE/dt = -GsynE/Tsyn : siemens
Is:ampere
Cm:farad
gl:siemens
El:volt
a:siemens
tau_w:second
Dt:volt
Vt:volt
Ee:volt
Ei:volt
Tsyn:second
'''

################### For A and B network ############################
# Population 1 and 3 : FS
G1 = NeuronGroup(N1, eqs, threshold='v > -47.5*mV', reset='v = -65*mV', refractory='5*ms', method
    ='heun') #same for G3

# Population 2 and 4 : RS
G2 = NeuronGroup(N2, eqs, threshold='v > -40.0*mV', reset='v = -65*mV; w += b2', refractory='5*ms
    ',  method='heun') #same for G' with w += b4'
```

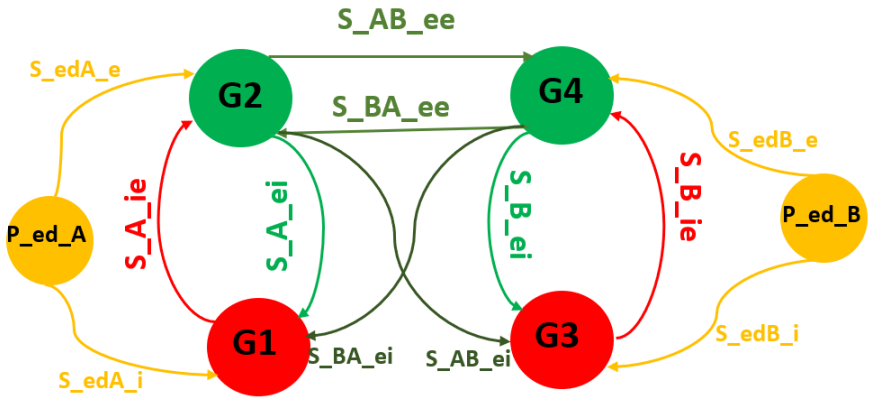


Figure 5.1: Possible synaptic connections for two columns within the AdEx

Synapses are composed of two **NeuronGroup** with a direction, in the arguments, the first group is the message emitter (a **NeuronGroup** or **PoissonGroup**) and the second is the receiver as displayed in the Fig. 5.1 below:

Moreover in the **connect** synapses' function, using 'i' and 'j' clearly define the connection relationship of a simple network where i refers to the source neuron index, and j to the target neuron index with a probability of connection, prbc ≡ 5%. Here, we voluntarily get rid of the autotapses, the synaptic connection between the same neuron. The *on_pre* keyword defines what happens when a presynaptic spike arrives at a synapse, here at a group of either inhibitory or excitatory synapses.

In addition to ordinary differential equations, Brian allows to introduce random noise by specifying a stochastic differential equation. For generating spikes according to a Poisson point process (By definition, a Poisson point process has the property that the number of points in a bounded region of the process's underlying space is a Poisson-distributed random variable), **PoissonGroup** can be used. Indeed, this is only a source of input to a neuron (i.e., the individually generated spikes are not important, just their impact on the target cell is). In the code, we create two groups of 8000 independent spikes and both groups fire at 0.8 Hz.

```
# external drive-------------------------------------------------
P_edA=PoissonGroup(8000, .8*Hz)
P_edB=PoissonGroup(8000, .8*Hz)

Qi=5.0*nS
Qe=1.0*nS
#Synapses from neuronal population to another within the same column
S_12 = Synapses(G1, G2, on_pre='GsynI_post+=Qi')
S_12.connect('i!=j', p=prbC)
  ...
S_44 = Synapses(G4, G4, on_pre='GsynE_post+=Qe')
S_44.connect('i!=j', p=prbC)

#Synapses from external drive to both populations of the same column
S_ed_inA = Synapses(P_edA, G1, on_pre='GsynE_post+=Qe')
S_ed_inA.connect(p=prbC)
...
S_ed_exB = Synapses(P_edB, G4, on_pre='GsynE_post+=Qe')
S_ed_exB.connect(p=prbC)

#Synaptic Connection from A (G1-G2) to  B (G3-G4) synapses

S_BAei = Synapses(G2, G3, on_pre='GsynE_post+=Qe')
S_BAei.connect('i!=j',p=prbC)
S_BAee = Synapses(G2, G4, on_pre='GsynE_post+=Qe')
S_BAee.connect('i!=j',p=prbC)

#Synaptic Connection from B (G3-G4) to A (G1-G2) synapses
S_ABei = Synapses(G4, G1, on_pre='GsynE_post+=Qe')
S_ABei.connect('i!=j',p=prbC)
S_ABee = Synapses(G4, G2, on_pre='GsynE_post+=Qe')
S_ABee.connect('i!=j',p=prbC)
```

5.2.2 Recording during a simulation

Recording variables during a simulation is done with "monitor" objects. Specifically, spikes are recorded with **SpikeMonitor**, the time evolution of variables with **StateMonitor** and the firing rate of a neurons'population with **PopulationRateMonitor**. By default, a **SpikeMonitor** only records the time of the spike and the index of the neuron that spiked, which is not the most relevant measurement in a neural network study.

```
M1G1 = SpikeMonitor(G1)
FRG1 = PopulationRateMonitor(G1)
 ...
M1G4 = SpikeMonitor(G4)
FRG4 = PopulationRateMonitor(G4)
```

5.2.3 Running a simulation

In order to run a simulation, we simply call the **run** function.

```
#######################################################
#Run the simulation

print('--##Start simulation##--')
run(duration)
print('--##End simulation##--')

#######################################################
#Prepare recorded data

#organize arrays for raster plots:
RasG1 = np.array([M1G1.t/ms, [i+N2 for i in M1G1.i]])
RasG2 = np.array([M1G2.t/ms, M1G2.i])

#Calculate population firing rate :

#function for binning:
def bin_array(array, BIN, time_array):
    N0 = int(BIN/(time_array[1]-time_array[0]))
    N1 = int((time_array[-1]-time_array[0])/BIN)
    return array[:N0*N1].reshape((N1,N0)).mean(axis=1)


BIN=5
time_array = np.arange(int(TotTime/DT))*DT
LfrG1=np.array(FRG1.rate/Hz)
TimBinned,popRateG1=bin_array(time_array, BIN, time_array),bin_array(LfrG1, BIN, time_array)
...
LfrG4=np.array(FRG4.rate/Hz)
TimBinned,popRateG4=bin_array(time_array, BIN, time_array),bin_array(LfrG4, BIN, time_array)
```

After the simulation, the attributes i, t, num_spikes and count of the spike monitor are accessible. The i and t attributes give the array of neuron indices for each spikes and times of all the spikes (t/ms to match the unit used in the simulation) and plotted as single dots with time on the x-axis and neuron index on the y-value, we get is the standard "raster plot" used in neuroscience. This plot enables to visualize the number and location of each neuron which spikes from every single population during the time simulation and gives access as well to specific patterns (e.g. less dots on a few parts of the plots repeated as a band pattern over the time) which can be interpreted for the states identification.

Many times we use a method called data smoothing to make the data proper and qualitative for statistical analysis. During the smoking process we define a range also called bin (here BIN=5) and any data value within the range is made to fit into the bin. This is called the binning method. The bin_array function, takes the time_array of size 10^5 and reshape the 1D array passed, the first argument, as a 2D array with N0=50 columns and N1=1999 rows. The final binned array will be the means of each column with the last 50 elements removed. That is how we obtain the population firing rate, the mean of all the neurons' firing rates within the **Neurongroup**.

5.3 Mean-Field Structure

All the codes for fitting the parameters, computing and loading the transfer function template used as a parameter in the function below are displayed in appendix B.

5.3.1 Contributions of each population

For the numerical evaluation of the transfer function, we compute the integration of the derivative of the transfer function ν_{out} by the variability of the synaptic connections probability and the overall and sub-population within each neural group, in the code this function is named "MeanField_dN_dp". It only evaluates the trend (the time derivative of all variables) at a given point in time and considers the possibility of having more populations, under study (cf. section 6.2). All the parameters are displayed in the table A.2.

First and foremost, we have to initialize the Runge Kutta fourth order (RK4) or Euler method of integration for the second order partial derivatives that takes three (or six) initial values for x0. Assigned values correspond to the initial state of the system. There are an infinite number of ways to initialize it, some of which converges to acceptable values, and some which explode. In our study case, the system is initialized close to what has been "measured" in the network for ν_{e_0} and ν_{i_0}, and for the adaptationw_0, it is initialized at the equilibrium point b*ν_e (point towards which, for a constant ν_e, the adaptation converges):

1- One *cortical column* Mean-Field model

- x0[0] ≡ ν_{e_0} = 60 (mV)
- x0[1] ≡ ν_{i_0} = 700 (mV)
- x0[6] ≡ w_0 = x0[0]*b (pA)

2- Two *cortical columns* Interconnected Mean-Field models (Different values are affected to the same population type in order to avoid a forced symmetry)

- x0[0] and x0[1] ≡ ν_{e_0} = 60 (mV) and 74 (mV)
- x0[2] and x0[3] ≡ ν_{i_0} = 700 (mV) and 840 (mV)
- x0[20] and x0[21] ≡ w_0 = x0[0]*b (pA) or x0[1]*b (pA)

Then, here are the main steps to compute the derivatives and and therefore the evaluation of ν_{out} (during the numerical integration):

1. Collect the inhibitory and excitatory contributions (including the Poisson drive): Transfer functions are not sensitive to the input frequencies but more to frequencies at given population sizes, which means the excitatory and inhibitory input frequencies ν_{0_e} and ν_{0_i} are calibrated to population sizes of N0_e = 8000 and N0_i = 2000 neurons, respectively. The number of sub-populations numb_sub_exc and numb_sub_inh are assumed to be evenly distributed.

```
for i in range(numb_sub_exc):
    Ntot_exc+=N[i]
for i in range(numb_sub_inh):
    Ntot_inh+=N[i+numb_sub_exc] #writing convention
```

5.3.2 Individual Transfer function evaluation

2. Evaluate the transfer function (TF) of each population and their partial derivatives with respect to the probability of synaptic connection and the number of neurons in the sub-populations: First of all, we have to define a matrix, "vsec_vec" in the code, in order to handle the different parameters (cf. (5.5)) passed in the transfer function.

```
vsec_vec=np.zeros((numb_var,numb_v_TF+numb_w_TF))
```

As mentioned above, if we consider a smaller excitatory population of 4000 exciters instead while using the same transfer function, we would have to take the average frequency of that population divided by 2, because they are twice fewer. Indeed, in the article [25], whenever the input frequencies, ν_{0_e} and ν_{0_i}, appear in the Mean-Field's machinery, they are preceded by K_μ, the size of the population sending the signal multiplied by the connectivity between the latter and the receiver:

$$K_\mu = N_{\mu_{send}} * prbc(i \neq j) \tag{5.20}$$

So, in the function, what we do is considering that the transfer function takes as input weighted means μ_{ν_e} and μ_{ν_i}, i.e., normalized by K_μ :

$$\mu_{\nu_e} = \frac{\sum_{k=0}^{Ntot_e} p_k * N_k * \nu_{e_k}}{p_0 * N_{0_e}} = \frac{\sum_{k=0}^{Ntot_e} p_k * N_k * \nu_{e_k}}{K_{\mu_e}} \tag{5.21}$$

$$\mu_{\nu_i} = \frac{\sum_{k=1}^{Ntot_i} p_k * N_k * \nu_{i_k}}{p_0 * N_{0_i}} = \frac{\sum_{k=1}^{Ntot_i} p_k * N_k * \nu_{i_k}}{K_{\mu_i}} \tag{5.22}$$

The μ_{ν_e}, μ_{ν_i} and w (the noise as well as the synaptic input (cf. 5.7)) are added to the matrix vsec_vec by applying the normalization (cf. 5.21) .

```
for i in range(numb_var): #receiving stimuli
    for j in range(numb_sub_exc): #sending stimuli
        vsec_vec[i][0]+=x0[j]*N[j]/N0_e*p[i][j]/(5e-2)

    for j in range(numb_sub_inh): #sending stimuli
        vsec_vec[i][1]+=x0[j+numb_sub_exc]*N[j+numb_sub_exc]/N0_i*p[i][j+numb_sub_exc]/(5e
    -2)

for i in range(numb_adapt):
        vsec_vec[i][numb_v_TF]=x0[numb_var*numb_var+numb_var+i]

for i in range(numb_var): ## external drive
        vsec_vec[i][0]+=(ext_drive[i]+input_func)*p_pois[i]/(5e-2)*Ntot_exc/N0_e
```

5.3.3 Delay line

If we want to add some varying or constant delay to the spike transmission between a connection of two synaptic groups, this delay has to be subtracted to the overall simulation time, (meas_time), both divided by the time step in order to increment x0. The mean time minus the delay is simply the current time (loc_time), i.e, the exact value of the time simulation where we compute each transfer function arguments' values.

```
meas_time=int(t/tstep)

for i in range(numb_var):
    for j in range(numb_sub_exc): ## excitatory input
        loc_time=int(meas_time-int(delays[i][j]/tstep))
        vsec_vec[i][0]+=x_hist[j][loc_time]*N[j]/N0_e*p[i][j]/(5e-2)

    for j in range(numb_sub_inh): ## inhibitory input
        loc_time=int(meas_time-int(delays[i][j+numb_sub_exc]/tstep))
        vsec_vec[i][1]+=x_hist[j+numb_sub_exc][loc_time]*N[j+numb_sub_exc]/N0_i*p[i][j+
    numb_sub_exc]/(5e-2)

for i in range(numb_adapt):
    loc_time=int(meas_time-int(delays[i][i]/tstep))
    vsec_vec[i][numb_v_TF]=x_hist[numb_var*numb_var+numb_var+i][loc_time]
```

5.3.4 Multi partial derivatives

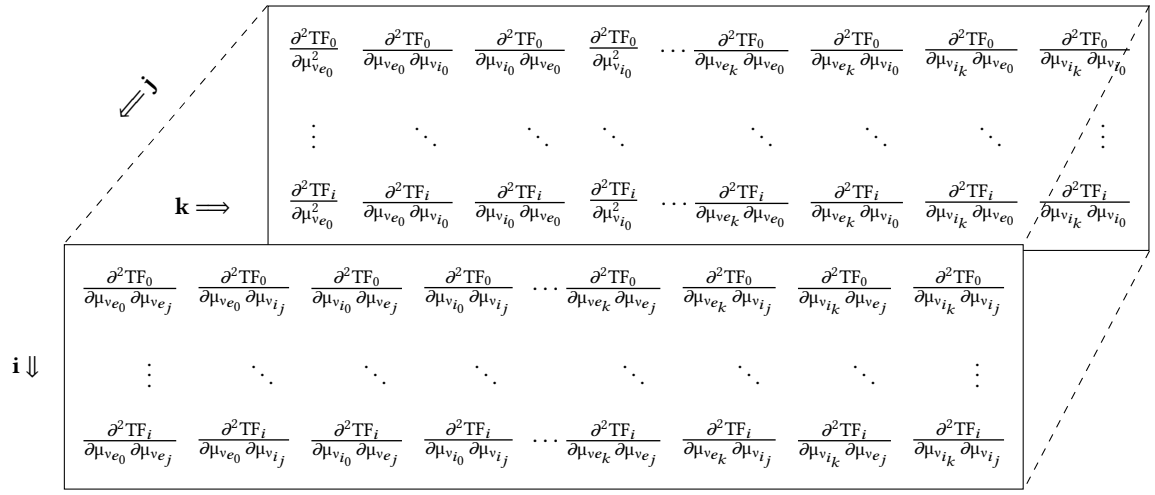
Thanks to the pointer on vsec_vec, we can map every argument of the transfer function and compute their partial derivatives of transfer function : ($\frac{\partial \mathrm{TF}_i}{\partial \mu_{v_{e_k}}}$, $\frac{\partial \mathrm{TF}_i}{\partial \mu_{v_{i_k}}}$ and $\frac{\partial w_i}{\partial w_k}$) :

```
for i in range(numb_var):
    Eval_TF[i]=F[i](*vsec_vec[i])
    for k in range(numb_sub_exc):
        Delta1[i][k]=deriv1b(F[i],vsec_vec[i],numb_v_TF,h,0)*N[k]/N0_e*p[i][k]/(5e-2)

#0 is taken because TF takes only excitatory and inhibitatory inputs, without discarding the
     subpopulation

    for k in range(numb_sub_inh):
        Delta1[i][k+numb_sub_exc]=deriv1b(F[i],vsec_vec[i],numb_v_TF,h,1)*N[k+numb_sub_exc]/
    N0_i*p[i][k+numb_sub_exc]/(5e-2)
```

Here is, the first order derivatives as a $numb_{var}$*$numb_{var}$ size matrix :

$$\Delta_{1_{ik}} = \begin{pmatrix} \frac{\partial \mathrm{TF}_0}{\partial \mu_{v_{e_0}}} & \cdots & \frac{\partial \mathrm{TF}_0}{\partial \mu_{v_{e_k}}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \mathrm{TF}_i}{\partial \mu_{v_{i_0}}} & \cdots & \frac{\partial \mathrm{TF}_i}{\partial \mu_{v_{i_k}}} \end{pmatrix} \tag{5.23}$$

And then we can compute the second order derivatives, $\Delta_{2_{ikj}} = \frac{\partial \Delta_{1_{ik}}}{\mu_{v_j}}$, which are a multidimensional array in order to calculate the cross correlation terms.




```
for i in range(numb_var):
    Eval_TF[i]=F[i](*vsec_vec[i])

  for k in range(numb_sub_exc):
      for j in range(numb_sub_exc):
          Delta2[i][k][j]=deriv2b(F[i],vsec_vec[i],numb_v_TF,h,0,0)*\
          N[k]/N0_e*N[j]/N0_e*p[i][k]/(5e-2)*p[i][j]/(5e-2)

      for j in range(numb_sub_inh):
          Delta2[i][k][j+numb_sub_exc]=deriv2b(F[i],vsec_vec[i],numb_v_TF,h,0,1)*\
          N[k]/N0_e*N[j+numb_sub_exc]/N0_i*p[i][k]/(5e-2)*p[i][j+numb_sub_exc]/(5e-2)

  for k in range(numb_sub_inh):
      for j in range(numb_sub_exc):
          Delta2[i][k+numb_sub_exc][j]=deriv2b(F[i],vsec_vec[i],numb_v_TF,h,1,0)*\
          N[k+numb_sub_exc]/N0_i*N[j]/N0_e*p[i][k+numb_sub_exc]/(5e-2)*p[i][j]/(5e-2)

      for j in range(numb_sub_inh):
          Delta2[i][k+numb_sub_exc][j+numb_sub_exc]=deriv2b(F[i],vsec_vec[i],numb_v_TF,h
    ,1,1)*\
          N[k+numb_sub_exc]/N0_i*N[j+numb_sub_exc]/N0_i*p[i][k+numb_sub_exc]/(5e-2)*\p[i][j+
    numb_sub_exc]/(5e-2)
```

5.3.5 Final Transfer function evaluation

3. Inject these partial derivatives in the formula of the temporal ones to integrate them and evaluate the transfer function.

```
for i in range(numb_var):
    deriv[i]=(Eval_TF[i]-x0[i])/T

    deriv[numb_var*i+i+numb_var]+=(Eval_TF[i]*(1./T-Eval_TF[i])/N[i])/T

    for j in range(numb_var):
        index=numb_var*i+j+numb_var

        for k in range(numb_var):
            deriv[i]+=(0.5*Delta2[i][j][k]*x0[numb_var*j+k+numb_var])/T #Second order added
            index_ik=numb_var*i+k+numb_var
            index_jk=numb_var*j+k+numb_var
            deriv[index]+=(x0[index_jk]*Delta1[i][k]+x0[index_ik]*Delta1[j][k])/T #First
    order added

            deriv[index]+=((Eval_TF[i]-x0[i])*(Eval_TF[j]-x0[j])-2*x0[index])/T


for i in range(numb_adapt):
    deriv[numb_var**2+numb_var+i]=-x0[numb_var**2+numb_var+i]/tauw[i]+b[i]*x0[i]+a[i]*(mu(x0
    ,params,i,t)-El[i])/tauw[i]

    return deriv,Eval_TF
```

We define here the location of the considered variable by the index and its derivatives:

$$\frac{\partial \mathrm{TF}_i}{\partial t} = \frac{1}{\mathrm{T}}\left((\mathrm{TF}_i - \nu_i) + \frac{1}{2}c_{2_{ijk}}\Delta_{2_{ikj}}\right) \tag{5.24}$$

$$\frac{\partial c_{2_{ijk}}}{\partial t} = \left(\frac{\mathrm{TF}_i(\frac{1}{\mathrm{T}} - \mathrm{TF}_i)}{\mathrm{N}_i} + (\mathrm{TF}_i - \nu_i) + c_{jk}\Delta_{2_{ijk}} + c_{ik}\Delta_{2_{ijk}} - 2 * c_{ij}\right) \tag{5.25}$$

$$\frac{\partial w_i}{\partial t} = \frac{w_i}{tau_{w_i}} b * \mathrm{TF}_i + a_i(\mu_{\mathrm{TF}}(\nu_i) - \mathrm{E}_{l_i}) \tag{5.26}$$

which corresponds to the Mean-field formalism for different populations (cf. eq. 5.8, where TF= ν)

5.3.6 Runge Kutta fourth order

Eventually,the derivatives and the Eval_TF are taken as arguments of the Runge Kutta fourth order numerical method which returns the transfer function values.At the end of the computation, we remove the last element of the time trace, *x_trace[-1]*, which correspond to the mean-field fixed point, thus we obtain 7 values for the transfer function of one *RS-FS* mean-field model and 22 for two interconnected *RS-FS* mean-field models (because of the 2^n correlation terms with n, the number of population). Notice $k_{1,2,3,4}$ of the code (xt here) are the arguments of the real $k_{1,2,3,4}$ in RK4).

```
def rk4_dN_dp_General(x0,params,time,func):

    p=params['numb_var']#variables needed to build arrays
    dx=np.zeros(p)
    deriv=np.zeros(p)
    xt = np.zeros(p)

    for i in range(p):
        dx[i]=deriv[i]=xt[i]=0

    deriv,TF=func(x0, time, params)

    for i in range(p):
        dx[i]=deriv[i] #x[i]=k1[i]
        xt[i]=x0[i]+0.5*deriv[i]*params['tstep']#argument for k2

    time+=params['tstep']/2;

    deriv,TF=func(xt, time, params)

    for i in range(p):
        dx[i]+=2*deriv[i] #x[i]=k1[i]+2*k2[i]
        xt[i]=x0[i]+0.5*deriv[i]*params['tstep'] #argument for k3

    deriv,TF=func(xt, time, params)

    for i in range(p):
        dx[i]+=2*deriv[i]
        xt[i]=x0[i]+deriv[i]*params['tstep']#argument for k4

    time+=params['tstep']/2;

    deriv,TF=func(xt, time, params)

    for i in range(p):
        dx[i]+=deriv[i] #x[i]=k1[i]+2*k2[i]
        xt[i]=x0[i]+deriv[i]*params['tstep']

    for i in range(p):
        x0[i]=x0[i]+dx[i]*params['tstep']/6.0

    return TF
```

We obtain for the interconnected model:

$$\nu_{out} = \begin{bmatrix} \mu_{\nu_{e1}} & \mu_{\nu_{e2}} & \mu_{\nu_{i1}} & \mu_{\nu_{i2}} \\ c_{e1e1} & c_{e1i1} & c_{e1e2} & c_{e1i2} \\ c_{i1e1} & c_{i1i1} & c_{i1e2} & c_{i1i2} \\ c_{e2e1} & c_{e2i1} & c_{e2e2} & c_{e2i2} \\ c_{i2e1} & c_{i2i1} & c_{i2e2} & c_{i2i2} \\ w_{e1} & w_{e2} & & \end{bmatrix} \tag{5.27}$$

6. Results

6.1 RS-FS Models

The parameters values for this simulation are referred in the table A.1. Nevertheless, the external drive was fixes at 2.5 Hz (value taken from the literature [25] for getting an asynchronous irregular state).The configuration under consideration is depicted on this simple sketch:

We consider here two populations: one excitatory and one inhibitory, the formalism thus describes the evolution of seven quantities: the two ν_e and ν_i [30], the variances *var_ee* and *var_ii* of the the excitatory and inhibitory population activity respectively and the covariance *cov_ei* and *cov_ie* between the excitatory and inhibitory population activities.By using the RK4 method while considering second order derivatives for mean-field integration, we determine numerically the fixed point of transfer function or ν_{out} which is the predicted firing rate of the neuron according to the transfer function $\nu_{out} = F_{\mu_V}((\nu_e, \nu_i), \sigma_V(\nu_e, \nu_i), \tau_V(\nu_e, \nu_i))$.

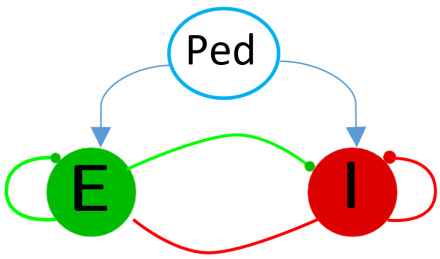


Figure 6.1: *RS-FS* model

ν_e	ν_i	$var_{\nu_e e} \equiv CC_{ee}$	$cov_{\nu_e i}$	$cov_{\nu_i e}$	$var_{\nu_i i} \equiv CC_{ii}$	w
0.673	7.181	0.0207	0.0364	0.0364	0.148	$4.035.10^{-11}$

Table 6.1: $\nu_{out} \equiv$ predicted output response

6.1.1 AdEx simulation

Thanks to the raster plot on Fig. 6.2, we can see that the neurons spikes randomly and sparsely. Moreover, there is no synchronous pattern repeated over the time. In order to prove, we are in front of asynchronous irregular states, the CV_{ISI} and the CC (c.f. subsection 5.1.1.1) for both population were computed and are displayed in Figs. 6.3 and 6.4. The bar plots show that all of the spiking neurons in both population spikes asynchronously because $CC \leq 0.01 \leq 0.1$ and with all $CV_{ISI} \leq 1$ for the excitatory population which means the spike trains are more regular than a Poisson Process with the same firing rate. Whereas the inhibitory population has most of its CV_{ISI} close to 1 but some are even < 1 which implies that some given spike trains are less regular than a Poisson process with the same firing rate. It seems that we witness an inhomogeneous Poisson Process or doubly stochastic Poisson process.

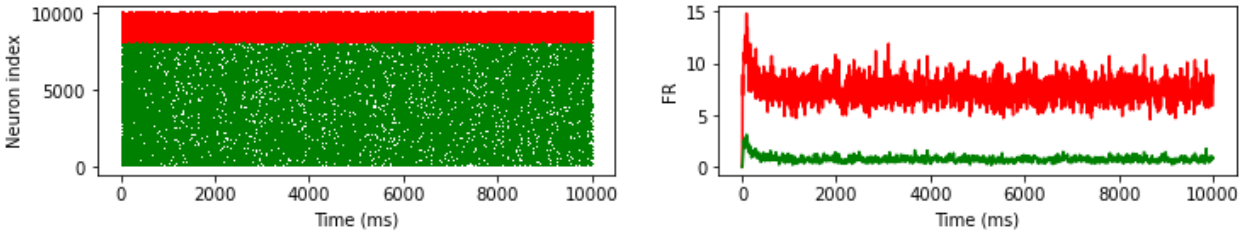


Figure 6.2: Raster plot and Time trace of Firing rates for the two population in the AdEx

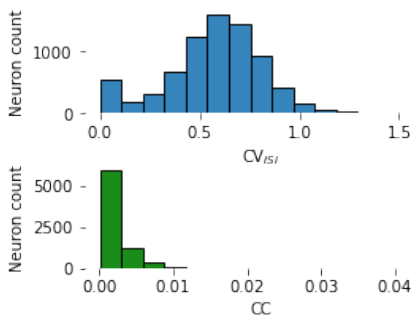


Figure 6.3: The CV_{ISI} and CC for RS cells

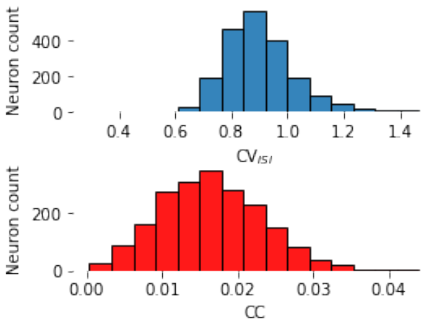


Figure 6.4: The CV_{ISI} and CC FS cells

6.1.2 Distribution

The Mean-Field Distribution is drawn using a Gaussian Kernel thanks to the means ν_e and ν_i as well as the standard deviations $\sigma_{ee} = \sqrt{var_{\nu_{ee}}}$ and $\sigma_{ii} = \sqrt{var_{\nu_{ii}}}$ evaluated beforehand.

The distribution of the excitatory population activity is rather well predicted by the formalism but it slightly underestimates the mean population activity above 1.7 Hz while it overestimates it for frequencies near 1 Hz. However, we can see a high discrepancy between the *AdEx* distribution and the mean-field one for the inhibitory population, the standard deviation is pretty high which narrows the Gaussian curve. The discrepancy is mostly observed for the lowest range, [3, 6.2] Hz, and the highest (above 7.5 Hz) spiking frequencies of the inhibitory cells. This phenomenon could be linked to the difference of noise implemented.

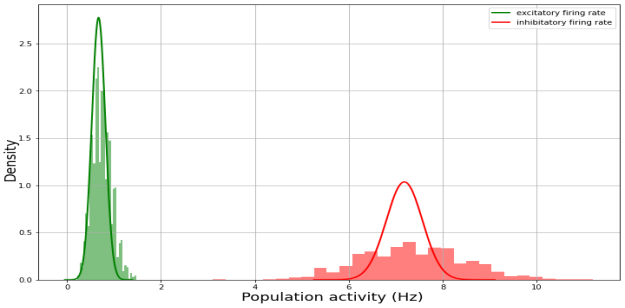


Figure 6.5: Firing rate distribution (curves for mean-field and filled bars for AdEx)

6.1.3 Macroscopic Noise

The Poisson process is used to model "microscopic" noise, but it is already implicitly "included" in the Mean-Field formalism, all firing rates expressing Poisson rates. Therefore, in order to account a higher level of definition, we introduce the *Ornstein–Uhlenbeck* process to add fluctuations centered around the mean-field's mean, it is a "macroscopic" noise here [25]. The choice of switching to Euler method is easier to explain while accounting the macroscopic noise. Incorporating noise into differential equations is always tricky, because the time step is a square root in the equations. A fourth order method like RK4 (Runge Kutta 4) would require more caution when including it, whereas it is easier with the Euler one. By reducing the time step, a level of accuracy comparable to that of RK4, in theory, can be achieved although the methods are not equivalent. For the mean-field equations (c.f. eq. 5.5 and 5.7), we used of the function called *Mean_Field_dn_dp_Hybrid_noise*, where noise is removed from second order terms for more stability. The discrepancy observed in the distributions are confirmed by the time traces of the Mean Firing rates but the two models are still displaying similar dynamics. For the transfer function time traces, they simply display the tendency of the *FS* population to fire more than the *RS* one.

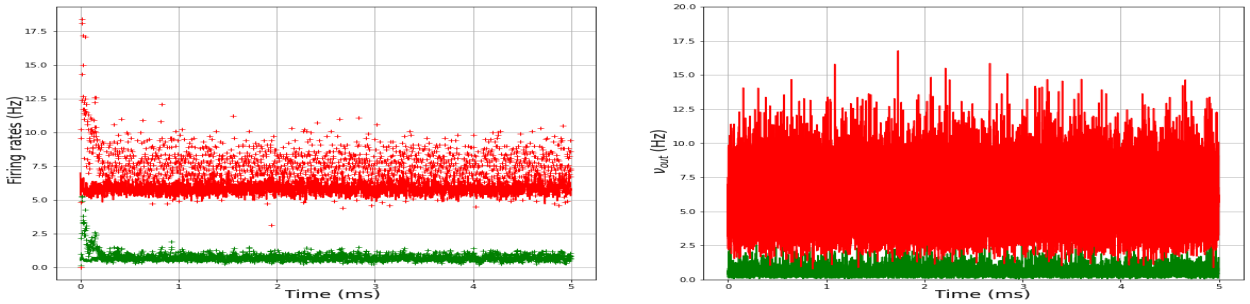


Figure 6.6: Time traces of the Firing rates and TF of *RS* and *FS* (resp. + and + are the mean-field prediction)

6.1.4 Effect of delay

From the literature [25], we know that longer is the delay for the sending and reception of a message between two synapses the more oscillations as we can see on Fig. 6.7 with a delay of 5 ms. This phenomenon of going back and forth from asynchronous irregular and synchronous regular/irregular states because of the delay and other parameters will be the main focus of the interconnected network simulation study.

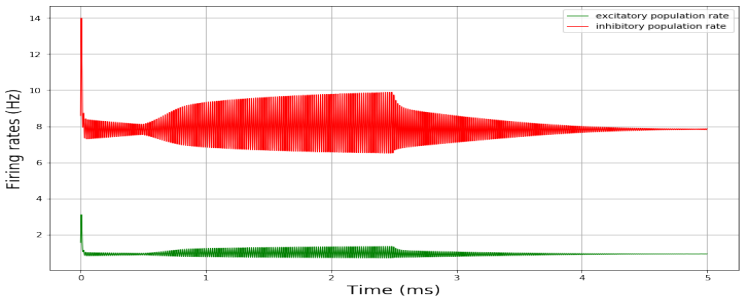


Figure 6.7: Mean-field model with a delay of 5ms

6.2 Interconnected Mean-Field models

The previously described configuration is drawn below:

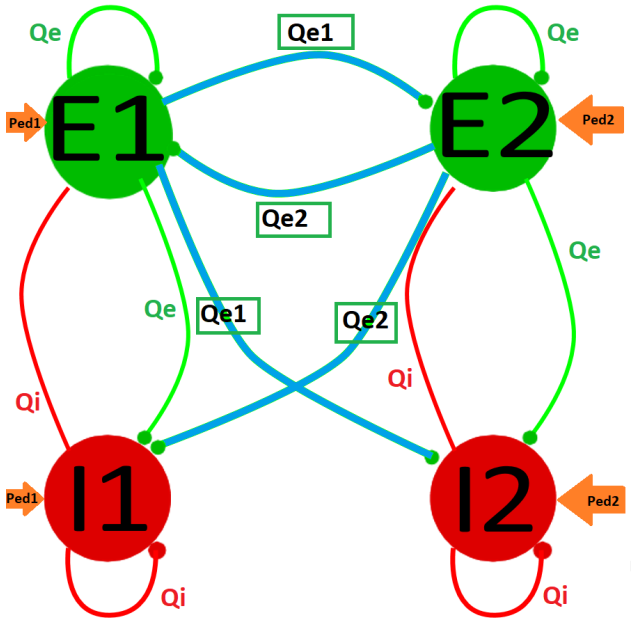


Figure 6.8: Interconnected columns configuration, these connections are subject to a delay, each column receive a different external drive

In this section, the study will undertake the model of two interconnected *MF* models. Usually, while focusing on the study of *cortical columns*, we look at a configuration where each network represents a brain area, so the connections between the two networks are exciting (but connect within one network exciters and inhibitors). Experimentally, excitatory connectivity has a longer reach in the cerebral cortex, than inhibitory connections. Therefore, if we take two cortical populations sufficiently away from each other, they exchange only excitatory connections, whereas if they are closer, they exchange both excitatory and inhibitory connections but never inhibitory only (except in special structures) as in the thalamus or cerebellum.

In order to simplify the problem, we are considering lateral connections to be only exciting, i.e., a given neuron receives "local" connections from its own network (excitatory and inhibitory) and "external" excitatory connections only, from the other network. The latter configuration is representative of the general case which display two essential parameters, the weight of the connection (between the 2 networks) and the delay (depending on the distance), quite important if we look at two distant networks for instance. Moreover, it has been observed that the presence of a delay promotes the emergence of oscillations which enables to understand the simulations of a large number of brain areas, which is our long-term goal. Consequently, one case will be assessed to study the relevant features role and the dynamics of the networks. The particular case corresponds to an extremely small value of the adaptation b, which corresponds to the state of an attentive, fully awake brain.

Here are displayed both, excitatory synaptic weights connection 6.2 and delay 6.3 matrices, as described earlier, we keep the same connection probability of 5%. However, the concerning the delay, it will be applied only to connection between different *cortical column* (i.e., the lateral connections).

		j columns RECEIVE			
		E1	**E2**	**I1**	**I2**
i lines SEND	**E1**	Qe	Qe_1	Qe	Qe_1
	E2	Qe_2	Qe	Qe_2	Qe
	I1	Qi	0	Qi	0
	I2	0	Qi	0	Qi

Table 6.2: Synapses connectivity matrix

		j columns RECEIVE			
		E1	**E2**	**I1**	**I2**
i lines SEND	**E1**	0	delay	0	delay
	E2	delay	0	delay	0
	I1	0	0	0	0
	I2	0	0	0	0

Table 6.3: Synapses connectivity delay matrix

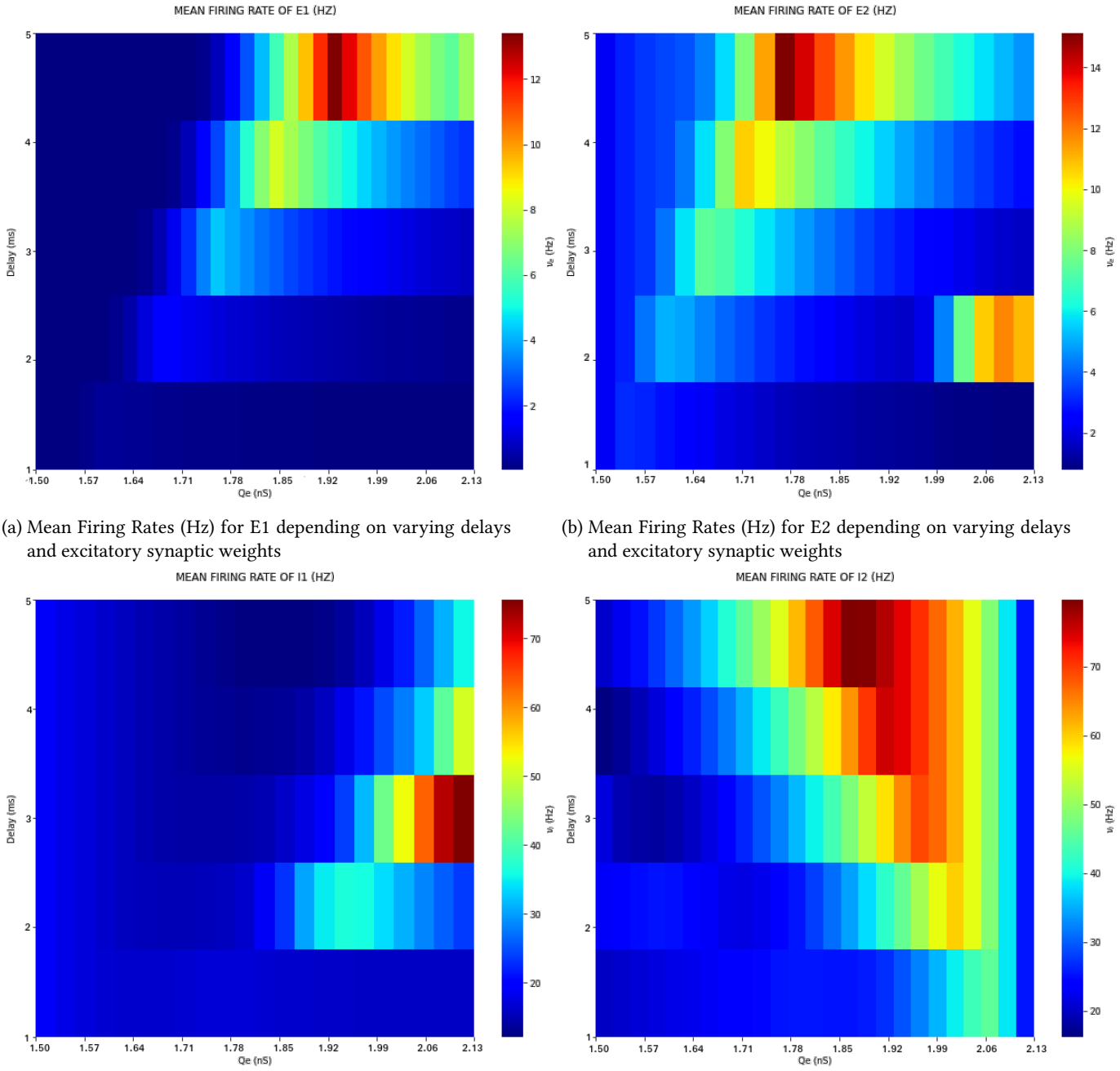
6.2.1 Simulation code

We would like to obtain two-dimensional graphics of the averages mean firing rates for each population as a function of two varying parameters, the delay (from 1ms (close *cortical columns*) to 5ms (the most distant *cortical columns*)) and the excitatory synaptic weights (defined for this model from 1.5 nS to 2.013 nS) which will only vary for the lateral connections. Therefore, I used another function to load the *MF* parameters, *load_transfer_functions_with_ALL_params_ij* where I could specify the i and j indices which corresponds to the four populations, defined in the parameters as a list. Therefore, the lateral connections will get varying Q_e (i.e., E1 would receive Qe_1 from itself and Qe_2 from E2) and the internal excitatory populations receive fixed synaptic weights Q_e as the inhibitors' ones with Q_i.

Thus, I used two nested loop with indexes, k and n, referring to the delay and the excitatory synaptic weights varying, these are iterables and enables matching this couple of values (Q_e,delay) to each corresponding mean firing rate of the population under focus. Here, we assigned shape[0]-1 because the array x_transp is already two dimensional and we only need the rows of this matrix.

```
mean_firing_rates_e1 = []
mean_firing_rates_e2 = []
mean_firing_rates_i1 = []
mean_firing_rates_i2 = []
delai=np.arange(1e-3,6e-3,1e-3)
qes=np.arange(1.6e-9,2.2e-9,7e-11)
for k, delay in enumerate(delai):
    for n, qe in enumerate(qes):

        ''' DEFINE THE INPUT FUNCTION '''
        par['qe'][0][1]=qe
        par['qe'][0][3]=qe
        par['qe'][1][0]=qe
        par['qe'][1][2]=qe
        par['delays'][0][1]=delay
        par['delays'][1][0]=delay
        par['delays'][1][2]=delay
        par['delays'][0][3]=delay

        x0=final_state.copy()
        t_trace=[]
        x_trace_RS=[]
        TF_trace_RS=[]
        x_trace_RS.append(x0.copy())

#########INITIATE THE PAST COORDINATES ##################
        delay_matrix=np.matrix(delay)
        max_delay=delay_matrix.max()

#### Associated number of time steps: Delay line ##
        steps=int(max_delay/params['tstep'])

        for i in range(steps):x_trace_RS.append(x0.copy())
############################################################
        time=0

        for i in np.arange(0,5,params['tstep']):
            par['x_hist']=np.transpose(x_trace_RS)
            TF_inst=rk4_dN_dp_General(x0,params,time,MeanField_dN_dp_delay)
            if (i%0.5==0):print('i = ', i, '\n')
            time+=params['tstep']
            t_trace.append(time)
            x_trace_RS.append(x0.copy())
            TF_trace_RS.append(TF_inst.copy())
        x_transp=np.transpose(x_trace_RS)
        mean_firing_rates_e1.append(x_transp[(k+n-1, x_transp.shape[0]-1), steps+1:])
        mean_firing_rates_e2.append(x_transp[(k+n, x_transp.shape[0]-1), steps+1:])
        mean_firing_rates_i1.append(x_transp[(k+n+1, x_transp.shape[0]-1), steps+1:])
        mean_firing_rates_i2.append(x_transp[(k+n+2, x_tranps.shape[0]-1), steps+1:])
```

Thereupon, with seaborn heatmaps, I was able to get these graphics displayed in Fig. 6.9.



(a) Mean Firing Rates (Hz) for E1 depending on varying delays and excitatory synaptic weights

(b) Mean Firing Rates (Hz) for E2 depending on varying delays and excitatory synaptic weights

(c) Mean Firing Rates (Hz) for I1 depending on varying delays and excitatory synaptic weights

(d) Mean Firing Rates (Hz) for I2 depending on varying delays and excitatory synaptic weights

Figure 6.9: Mean-Field Firing Rates dynamics with a small adaptation b

6.2.2 Criteria of states specifications

- The difference between limit cycle and the Up/Down oscillations: On one hand, for the Up/Down states, the minimum value is almost zero (the "down"), in fact it is zero + the external drive. On the other hand, the limit cycle oscillation has a minimum which must be above the zero + external drive.
- The Up/Down waves are actually determined by the noise that makes a switch between two metastable state. Therefore, the Up/Down waves must necessarily be much more variable than the limit cycle oscillations.
- Computing the standard deviation for each population, the separate time traces of E1, E2, I1 and I2. Concretely, at each time step (after the transient), it would be a matter of recording two statistics. It enables to distinguish fixed points from dynamic points and if the standard deviation is 0, the states are asynchronous irregular AI, if the value is higher bigger of "finite" size, there are time variations which should be qualitatively interpreted. Here an example for the first *cortical column* excitatory population:

```
mean_e1=0
std_e1=0

for i in (. . . ): #integration loop

mean_e1+=ve1
std_e1+=ve1**2

#output of the integration loop
mean_e1=mean_e1/number_of_time_steps #time average
std_e1=np.sqrt(std_e1/number_of_time_steps-mean_e1) # standard deviation over time sqrt(<X
    **2>-<X>**2)
```

6.2.2.1 Parameters and fixed points for bifurcation and phase planes

If we consider the equations 5.5 at the second order and write it with the fundamental hypothesis (cf. eq. 5.16):

$$m_\alpha^s(t) \equiv \langle \nu_\alpha(\mathrm{E}_\gamma')(t) \rangle \tag{6.1}$$

with $\langle . \rangle$ corresponding to the noise average. Therefore, at the mesoscopic level we have:

$$\overline{m_\mathrm{I}} = \mathrm{F}_e(\overline{m_\mathrm{E}}, \overline{m_\mathrm{I}}) \tag{6.2}$$
$$\overline{m_\mathrm{E}} = \mathrm{F}_i(\overline{m_\mathrm{I}}, \overline{m_\mathrm{E}}) \tag{6.3}$$

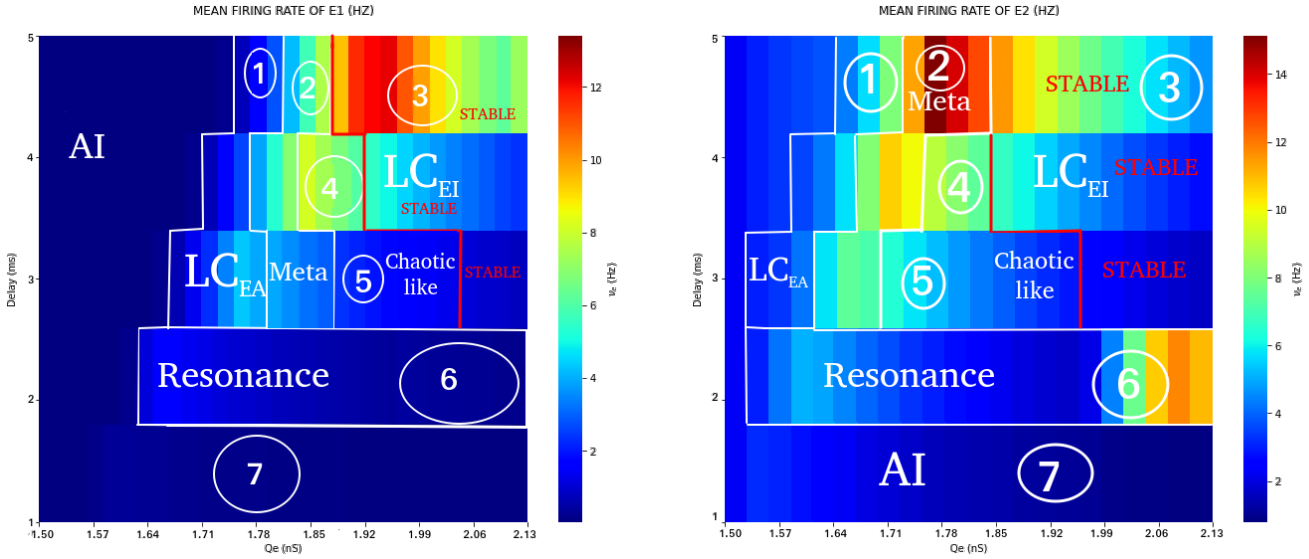
After solving the above equations 5.9 simultaneously, we obtain the values $\overline{m_{\mathrm{E}_0}}$ and $\overline{m_{\mathrm{I}_0}}$ at the fixed points. By neglecting the higher order components, we obtain the following locally linearized equation:

$$m_0^s(t) = \overline{m_0^s(t)} + \bar{\partial}(m_0^s(t)) \tag{6.4}$$
$$\partial m_0^s(t+1) = \mathrm{K}\partial m_0^s(t) \tag{6.5}$$

with $\mathrm{K}^\alpha = \Delta_{1_{ik}} = \frac{\partial \mathrm{TF}_i}{\partial \mu_{\nu_{\alpha_k}}}$, the jacobian matrix computed in the code section (α, γ and $s \in \{\mathrm{E},\mathrm{I}\}$. Therefore, the K_s and $\overline{m_s}$ will be the y and x-axis of the *phase planes*.

6.2.3 Bifurcation states map for E1 and E2

By gradually changing the delay and Q_e, we map out the state space of the system, depicted in the bifurcation diagrams in Fig 6.10. Bifurcations separate the state space into distinct regions of network states between which the system can transition from one to another. In our case, the dynamical states of our system depends on internal parameters changes to both sub-populations within each *cortical column*, which are directly affected by external driving sources, e.g., inputs from other neural populations such as the other *cortical column*. The frequencies range for both excitatory population display θ, α and β ranges. The two excitatory populations almost have the same dynamics except for a few areas which will be discussed further:

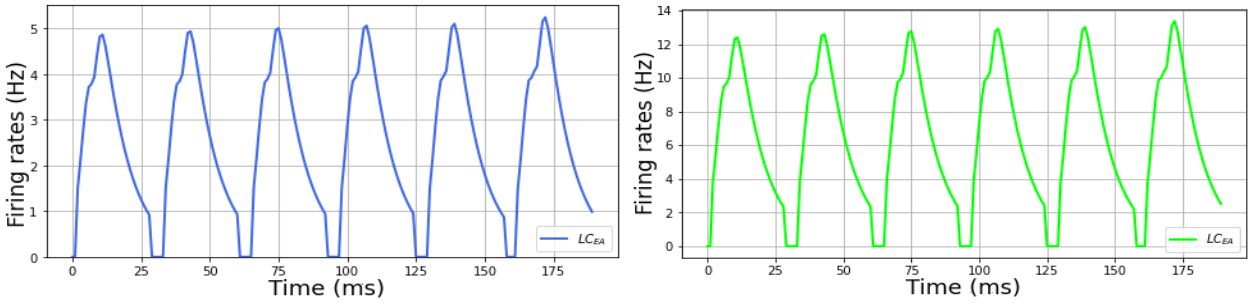


(a) Bifurcation states diagram for population E1

(b) Bifurcation states diagram for population E2

Figure 6.10: The different states observed for E1 & E2

1. Slow excitation-adaptation limit cycles (LC_{EA}) : These synchronous regular oscillations are a special type of limit cycle. They almost remain at the same frequency and amplitude but their minimum reach 0 Hz. These kind of slow oscillatory region labeled as LC_{EA} in the Figures 6.11a and 6.11b are caused by finite adaptation. The mean-firing rates for these slow oscillations range from 1 Hz to 2 Hz for E1 and 5 Hz to 7 for E2 and they seem to appear for rather low excitatory synaptic weight (1.64 nS to 1.78 nS for E1 and between 1.5 nS to 1.52 nS for E2) and need a higher delay to emerge at higher Q_e values.

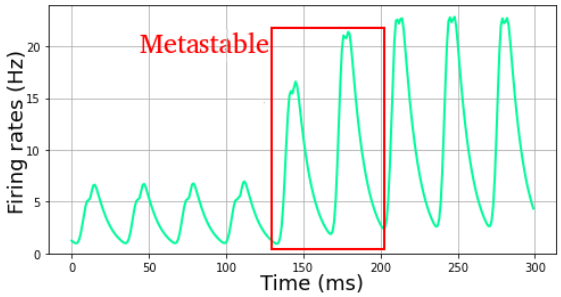


(a) LC_{EA} time trace for E1

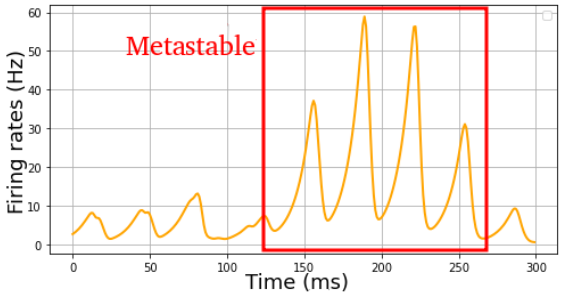
(b) LC_{EA} time trace for E2

Figure 6.11: States 1 for both E1 and E2

2. Meta-stable state: Synchronous irregular waves are bursting afterwards as shown in Figures 6.12a and 6.12b. The phase delay peaks hypothetically owing to the unstable equilibrium point that threads the cerebral cortex's limit cycles (State 3, 4 and 5). The excitatory population probably got the upper hand over the inhibitors in order to trigger a weightier activity and with a switch from stable to unstable waves.



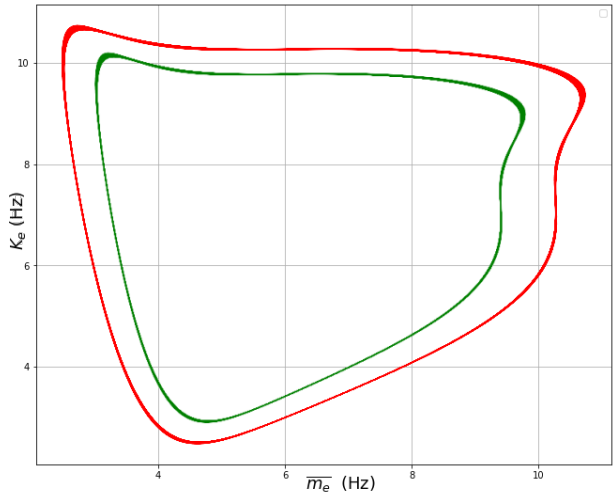
(a) Meta-stable state time trace for E1



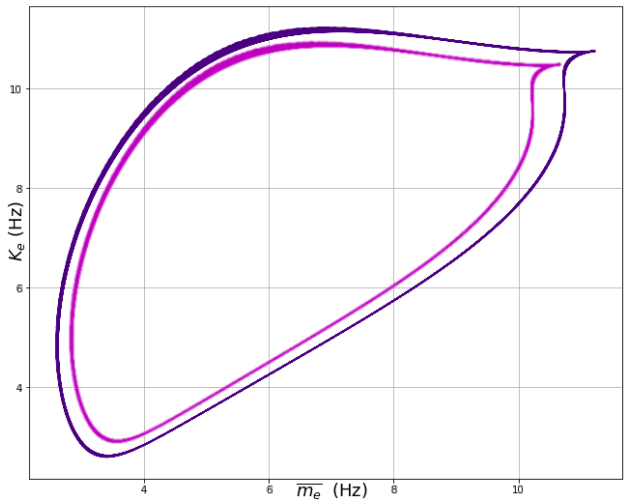
(b) Meta-stable state time trace for E2

Figure 6.12: States 2 for both E1 and E2

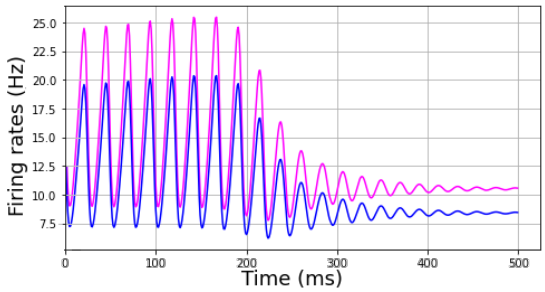
3. Fast excitation-inhibition limit cycles (LC_{EI}): The coupling of excitatory and inhibitory neurons gives rise to an oscillatory limit cycles LC_{EI} with an alternating activity between the two populations. In this case, the two limit cycles will keep periodically oscillating with long-term behavior (cf. Figures 6.13d and 6.13d). Figures 6.13a and 6.13b further demonstrates the *phase planes*. As the value of synaptic weight increases, the system dynamics vary a two limit cycles cohabitation [36].



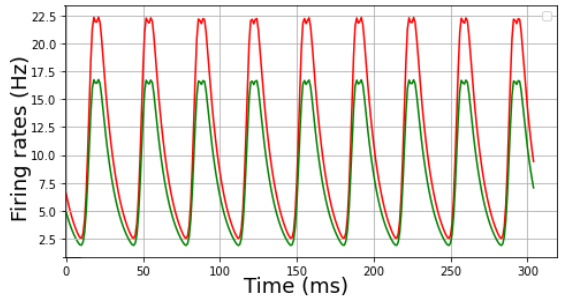
(a) Phase plane of two limit cycles in E1



(b) Phase plane of two limit cycles in E2



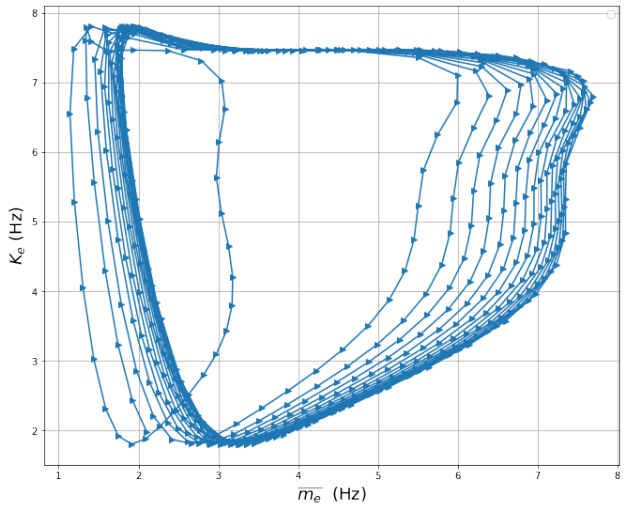
(c) Two stable limit cycles time trace in E1



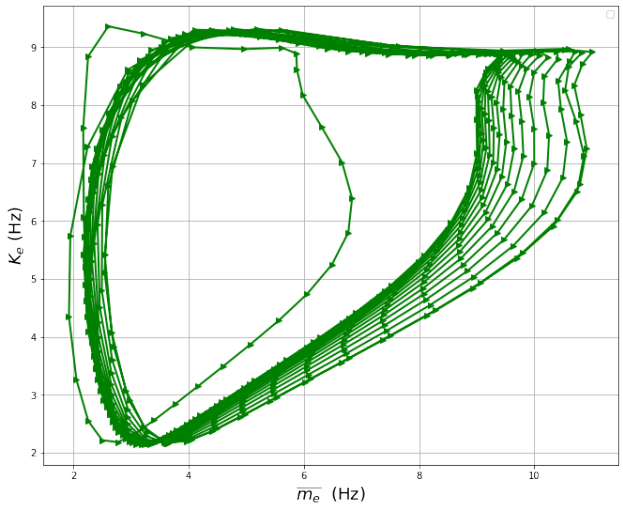
(d) Two stable limit cycle time trace in E1

Figure 6.13: State 3 for both E1 and E2

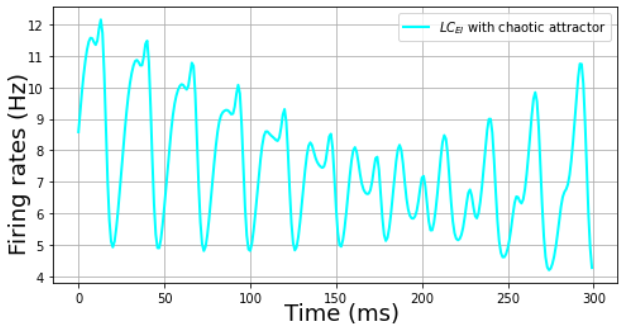
4. Chaotic-like states or Period doubling cascade: At 4 ms of delay, the dynamics of the simulations from a bit after 1.78 nS to 1.92 nS for E1 and from 1.71 nS to 1.85 nS for E2 exhibit period-doubling cascade, leading to dynamical transition from period to chaos. When the Q_e value increases to the range maximum mentioned above, the period of oscillation increases by 2^n with n the number of previous limit cycles [36]. Consequently, close to the limit's end, we find periodic orbits with period 16. In the *phase planes*, these periodic dynamical behaviors represent 16 limit cycles (see Figures 6.14a and 6.14b). As a result of the period-doubling cascade, the occurrence of complex periodic behaviors and chaotic dynamics are pinpointed in the time traces at 1.85 nS (cf. Fig. 6.14c) and 1.92 nS (c.f Fig. 6.14d).



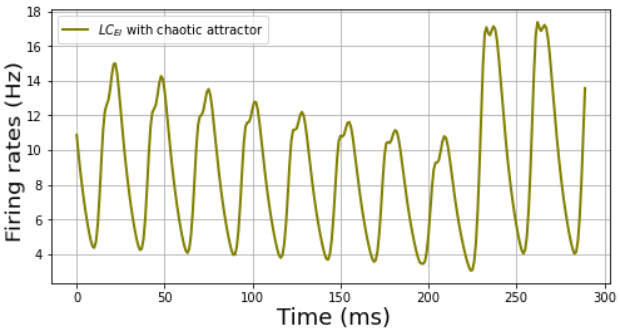
(a) *Phase plane* of period-double cascade in E1



(b) *Phase plane* of period-double cascade in E2



(c) Time trace breaking through a chaotic attractor in E1

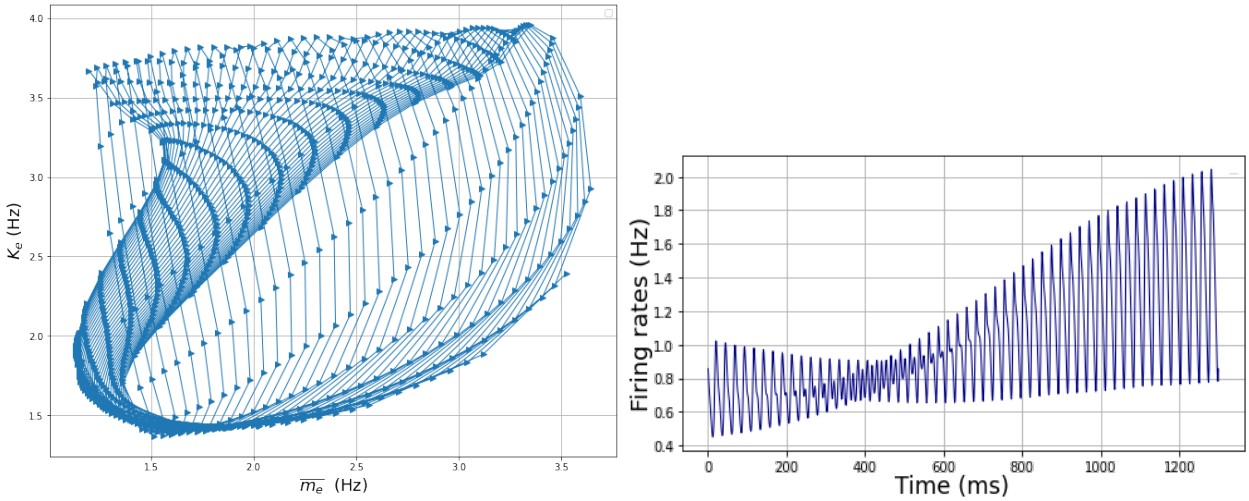


(d) Time trace breaking through a chaotic attractor in E2

Figure 6.14: State 4: From Limit cycle to chaos

5. Other Chaotic-like states ⟶ Spiral attractor and Slow fast cycles:

   - Population E1 : We observe in Figure 6.15a, the Shilnikov spiral chaos [37]. The transition to spiral chaos along the lines of the above scenario is usually preceded by either a cascade of period doubling bifurcations or the collapse of a two-dimensional torus. Indeed, our spiral follows a period-doubling cascade here, due to two nested unstable limit cycles which start and finish at the same attractor inside each other cycles. The spiral triggers homoclonic orbits to trap the other ends of the two dimensional unstable and one dimensional stable manifolds of the saddle equilibrium. In our high dimension space, bifurcations can occur, producing complicated, possibly chaotic dynamics as witnessed in Figure 6.15b.

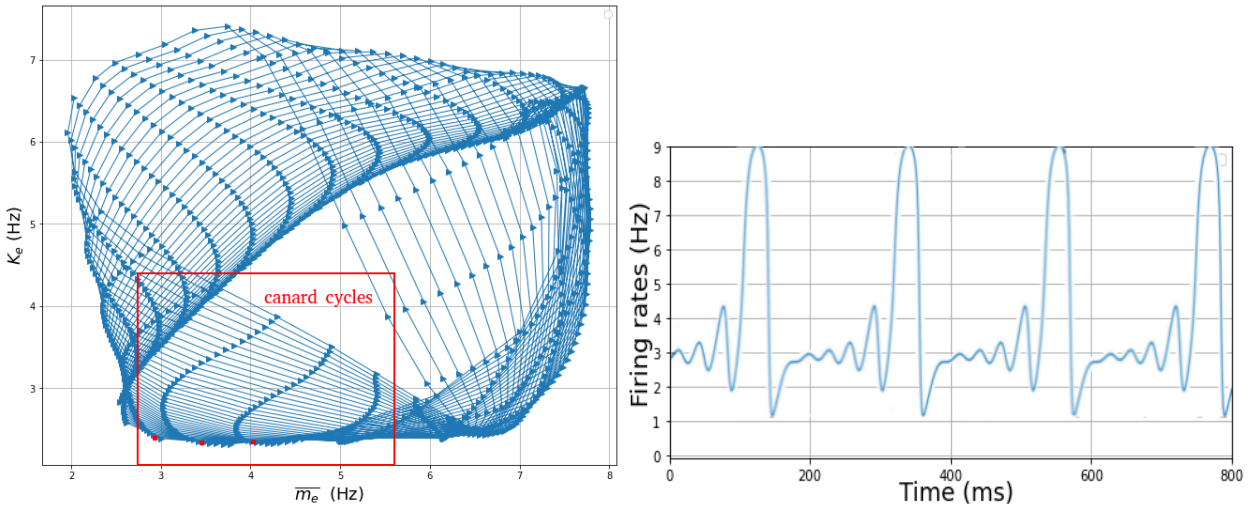


(a) Spiral chaos after period doubling cascade

(b) Time trace of the chaotic behaviour with strange attractor

Figure 6.15: State 5: Spiral chaotic attractor

   - Population E2 : In Figure 6.16a, we detect so-called 1 or 2–canard cycles consisting of fast orbits, turning around the Möbius strip [38]. The corresponding time trace in fig. 6.16b display mixed mode oscillations corresponding to switches between small amplitude oscillations and relaxation oscillations and usually appear owing to slow passage through a delayed Hopf bifurcation (i.e., a local bifurcation in which a fixed point of a dynamical system loses stability). The firing neurons produce high frequency oscillations and bursting activity due to the association of lower firing neurons [39]. This transition, also called canard explosion, occurs through a sequence of canard cycles which can be asymptotically stable, but they are hard to observe in an experiment or simulation because of sensitivity to the control of parameters and also because of sensitivity to noise.

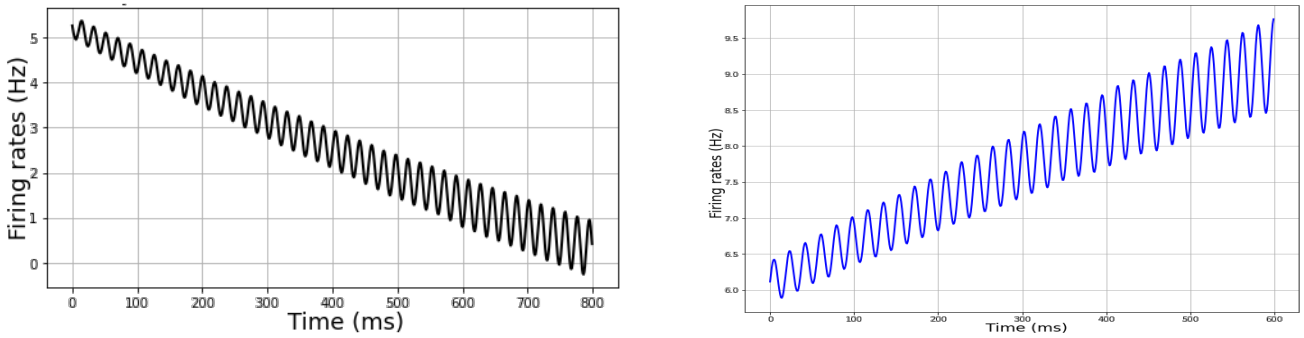


(a) *Phase plane* of spiral chaos after period doubling cascade

(b) Time trace of Mixed Mode Oscillations

Figure 6.16: State 5 for E2

6. θ-Resonance : Activated neurons, which fire at an earlier phase for delayed connections at 2 ms, strengthen outward connections (cf. Fig. 6.17b) for the population E2 with forward directed oscillations. Neurons given lower external inputs, like in the population E1, with reverse directed oscillations, do the opposite, weakening outputs as pictured in Fig. 6.17a. For the both population, the θ oscillations could be coupled with γ ones due to the small amplitude.

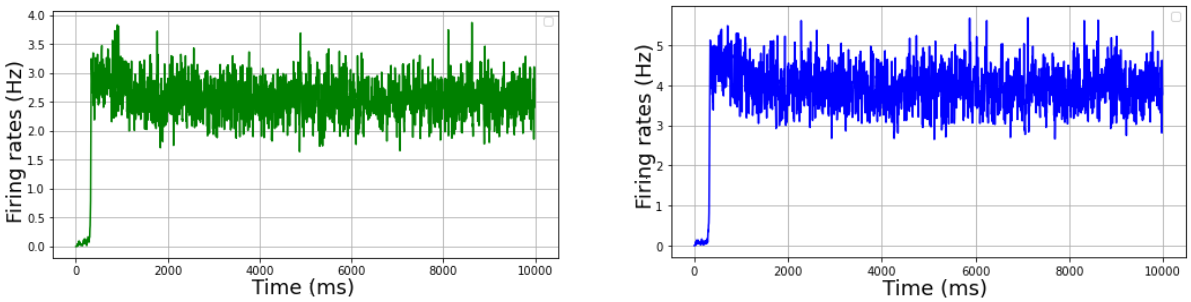


(a) Time trace of θ-Resonance oscillations for E1

(b) Time trace of θ-Resonance oscillations for E2

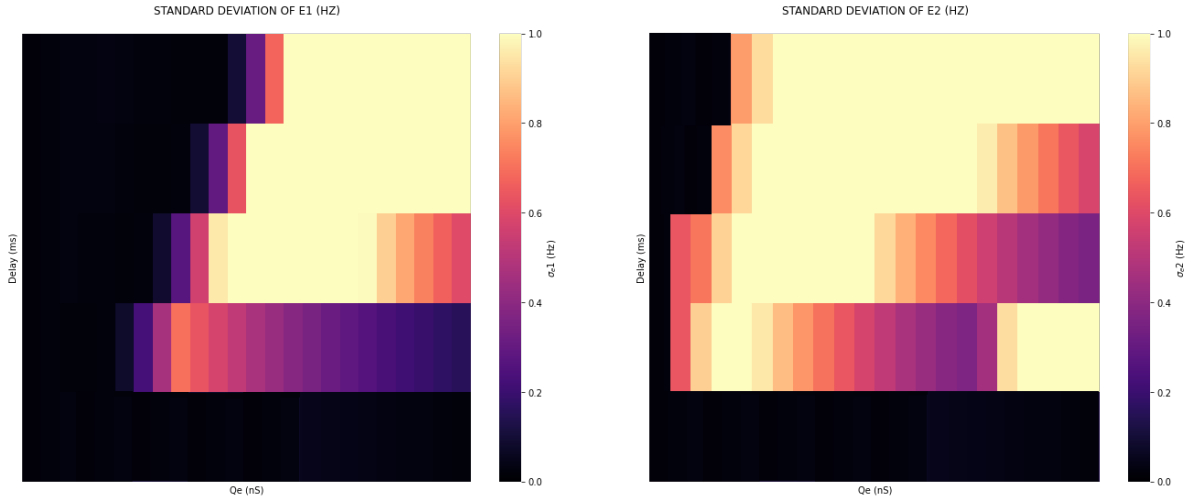
Figure 6.17: State 6 for E1 and E2

7. AI States :For both populations, we can easily notice, the constant range of spiking over the time with asynchronous patterns. We determined them thanks to the standard deviation for each population, if it equals 0, we are in asynchronous irregular state in Fig. 6.18c and 6.18d.



(a) Time trace of AI for E1

(b) Time trace of AI for E2



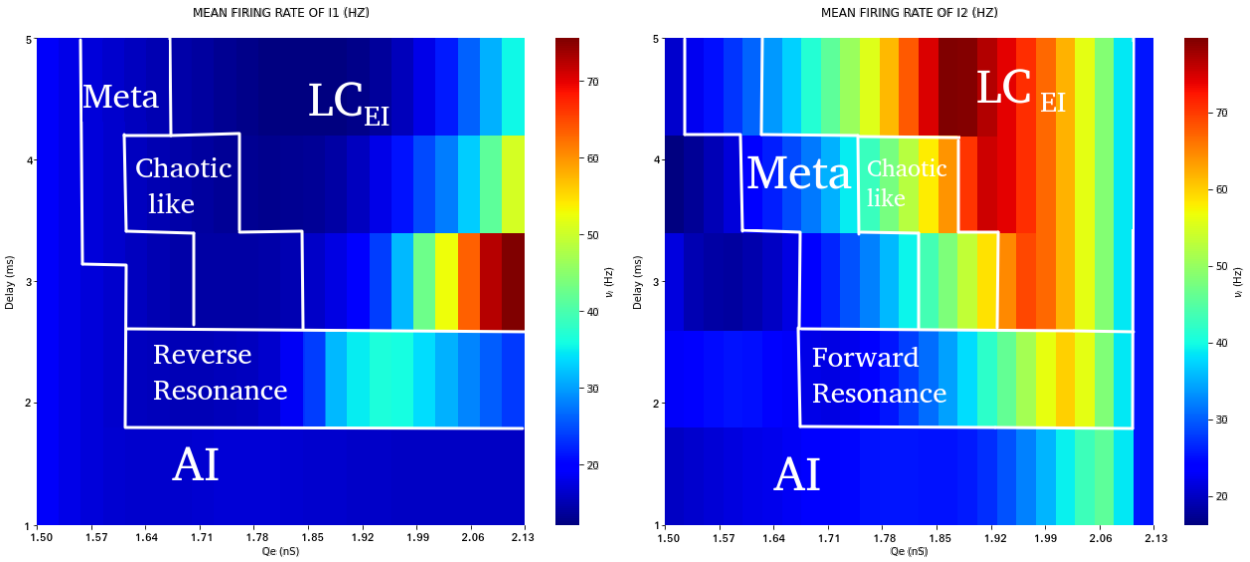
(c) Standard deviation for E1

(d) Standard deviation for E2

Figure 6.18: State 7 for E1 and E2

6.2.4 Bifurcation states map for I1 and I2

Inhibitory populations follow approximately the same states and their position according to the delays and Q_e. However we perceive that LC_{EA} are absent from the simulations, which could be explained by the fact that adaptation does not produce such slow limit cycles while being coupled with inhibition.Moreover, we are catching the sight of a strange phenomenon at the highest Q_e value for the population I2, there is a drop of the mean firing rate from the ɣ frequencies range to the β one.



(a) Bifurcation states diagram for population I1

(b) Bifurcation states diagram for population I2

Figure 6.19: The Bifurcations states map for I1 & I2

7. Discussions

7.1 Discussions and Future Prospects

For the interconnected *AdEx*, because of the time consuming simulation for these models and a lack of an overall time, I decided to take one couple of values (Q_e, delay) for each state which appeared in the corresponding mean-field models and simulate the firing rates for these values, while keeping the same configuration for the two *cortical columns* networks. However, I could not make an accurate analogy between the two models without doing the bifurcations states map and the whole analysis built in Section 6.2. Nevertheless, we can still draw relevant conclusions.

7.1.1 Results Discussion

Mean field models seem to simulate asynchronous irregular states very well, which is not surprising since *MF* were invented to specifically model these states (considered to be the most representative ones of the cortical main activity).

The mean field model has very interesting dynamics. Indeed, there are transitions between asynchronous irregular (AI) states to LC_{EA} ones whereas LC_{EA} are rather uncommon to be seen for such a low value of adaptation and normally precede the Up states (in the Up& Down states). Here, they lead to meta-stable states, there are semi-transient signals in the brain that persist for a while and are different than the usual equilibrium state (unstable and stable phases switches over the time) which thereupon trigger chaotic-like phenomena (around 3 and 4 ms of delay), probably displaying the the mean-field limit where the network induces the destabilization of attractors and chaotic oscillations [40]. **Then, appears to be a relationship between the delay and the stabilization of the system, the higher the delay is the more rapidly there is a transition from chaos to** LC_{EI} **for lower** Q_e **values**.

A delay lower than 3 ms, regardless of the value of the Q_e, seems too small to induce limit cycles, but **at 2 ms an interesting phenomenon is observed for fairly low to high** Q_e **values called the** θ**-auto-resonance**. Sequences of place cells activation occurring during spatial exploration are replayed during subsequent θ oscillations, and surprisingly, these replay events can occur in either the forward (being more prevalent during sleep) or reverse (the network is reactivated by oscillating input) direction within the hippocampus, an elaboration of the edges of the cerebral cortex, θ waves play a central role in the function of place cells activation, which encode spatial and contextual information [41]. Eventually, we could hypothetically stated that the population could correspond to two *cortical column* parts in different learning processes.

The new relation between delay and Q_e **synaptic has been discovered : The dominant inhibition state transition, determined by the g ratio, seems delay dependent when delay** $\geq$ **5 ms**. The smaller this ratio is the smaller the dominant inhibition state area is on the bifurcation map. It normally decreases with the increase of Q_e. The second *cortical column* received higher initial mean inputs which would usually lead to more inhibition as we witness in the tables 7.1 and 7.2 except for a delay of 5ms. **Hence, a long delay could hypothetically switch the dominant population thus the transition between states**

	Delay (ms)	g=$\frac{Q_i}{Q_e}$ (nS)
E1	5	2.75
	4	2.6
	3	2.43
E2	5	2.7
	4	2.7
	3	2.56

Table 7.1: Comparative table of g for E1 and E2 (smallest value ≡ smaller dominant inhibition area)

	Delay (ms)	g=$\frac{Q_i}{Q_e}$ (nS)
I1	5	2.99
	4	2.84
	3	2.7
I2	5	3.05
	4	2.65
	3	2.65

Table 7.2: Comparative table of g for I1 and I2 (smallest value ≡ smaller dominant inhibition area)

7.1.2 Limits of Mean-fields

For a small adaptation value, the overall behaviour seems not so close to the *spiking neural networks* one based on some isolated simulation I made and the literature (cf. [42]), there is the apparition of slow oscillations states for the all populations which does not appear with the *SNN* models. We could suspect that the oscillatory states are where the steady-state approximations that are used to construct the mean-field model break down due to the fast temporal dynamics in this state. Hence, both models have notable differences between the oscillatory regions [42]. Sharp transitions between states cause transient effects that are visible as ringing oscillations in the population firing rate, typically observed in simulations of *AdEx* (such as in Fig 6.2).

MF seems to model the behavior of a *cortical column* the same way as the *AdEx* as observed on the section 6.1, but when we scale up we move away from the behaviors observed for the *AdEx* ones, which suggests that models such as the connectome of the Virtual brain may induce errors in simulations of the cerebral cortex's states (with serious consequences if used for diagnostic purposes as it is intended to).

No model is able to describe the dynamics of second-order statistics at the network level. The Markov approach (cf. eq. 5.14) is directly constrained by the population activity correlation's fine structure during asynchronous irregular states and the minimal bin size to capture network dynamics.The master equation (cf. 5.19) directly relies on the neuron transfer function, which can be chosen according to the desired network model. For conductance-based models, as ours, no exact solution of the equation can be found. We also have to seek approximations to obtain the transfer function. In this case, the discrepancies observed for the distributions in Fig. 6.5 in section 6.1 are likely due to residual correlations caused by finite size effects and the phenomenological (semi-analytical) transfer function. Indeed, for activity in small networks like one *cortical column* or even the two interconnected *cortical columns* ones, the diffusion approximation is no longer legitimate, and the corresponding transfer function leads to incorrect predictions.

Moreover, the use of so called "tricks" in the Mean-field model in order to produce *AdEx* results are commonly used for forcing the system to be asymmetrical by changing the experimental input values or correcting abnormal values obtained by the integration methods. These values or the symmetry of the system are a sign of the model explosion, i.e., we have exceeded the validity scope of the Mean-field models, whose boundaries are still unknown. Moreover, the *Ornstein–Uhlenbeck*'s choice for the macroscopic noise is a sparing choice, it is the simplest process as a way of get standardized noise, directly inspired by Volo, Romagnoni, Capone, *et al.* [25]'s method. One might wonder if this choice is indeed justified, and how to properly establish the noise parameters, the ones used here, being absolutely arbitrary. As a consequence, these methods spread to the next person who will face the model (e.g. myself and people who are using the mean-field in The Virtual Brain). The simulations are made in a very controlled framework, compared to the Virtual brain, where besides the high number of nodes, there are also shady parts in the codes that not many seem fully aware of. This is due to the models and the domain itself which present many grey areas such as knowledge, from a purely physiological point of view, of the brain and its dynamics. Consequently, trying to model the most complex and unknown object without proper fundamentals and proofs will automatically lead to error propagation over the time and within the application field. As a result, the research papers are mostly theoretical.

7.1.3 Improvement Approaches

One possibility, the most direct, is to perform simulations with two interconnected *AdEx* networks under the same conditions as the two interconnected *MF* to construct the bifurcation states map properly, interpret the different states through the same methods and the model's quantification tools as mentioned in subsection 5.1.1.1. This will confirm that a two interconnected mean-field system continues to be "representative" of the underlying scales or not.

Accordingly to the small adaptation limit, the effects of the fluctuations in the dynamics of adaptation (as a result of firing rate fluctuations) becomes crucial at small τ_w (ref. section 6.1 with τ_w= 500 ms and section 6.2 τ_w = 1 s). While we are here interested in slow dynamics of adaptation (i.e. $\tau_w \geq$ 200ms) it would be interesting to consider the effect of fluctuation on adaptation, for example deriving a second order mean field also in the variable of adaptation following Buice, Cowan, and Chow [43] work. Notice that the role of fluctuations in adaptation current for small τ_w with respect to τ_m has been reported also at the level of the *AdEx* neuron transfer function by Hertäg, Durstewitz, and Brunel [44]. Furthermore, the poor reproduction of the oscillations in our model is likely due to the use of an exponentially-decaying linear response function instead of a damped oscillation which would be a better approximation of the true response as proved by Augustin, Ladenbauer, Baumann, *et al.* [45]. This constitutes a possible improvement of the work.

For now on, no real comparison between the two models under focus has been made on a larger scale (e.g. SpiNNaker [46] [I] and BrainScaleS [II] are large-scale neuromorphic machines composed of millions of connected *cortical columns* modeled by *AdEx*). However, the simulations at conventional supercomputers typical run factors of 1000 slower than biology and cannot access the vastly different timescales involved in learning and development ranging from milliseconds to years [47], [48]. Moreover, the implementation of the mean-field version is still in process but would be an important tool to check the mean-field validity scope on the brain scale.

Another possible approach could be to use a firing rate-based model instead of the spike-based one, everybody is using, because existing methods captures spikes indirectly and neurons communicate with spikes (cf. section 4.1.1 but not the firing rates, yet. In other words: do rates have a causal value in the dynamics of neural networks? The problem is, spikes have a causal role (e.g. the notion of a postsynaptic potential), but firing rates, most of the time are averaged like in the mean-field, with no causal nature. In fact, given that a firing rate is an observable that is defined on top of another observable, spikes, the question reduces to a more formal question, about relations between two formal systems: can a spike-based model , like *spiking neural network* model, of the brain be approximated by a firing rate-model, like *MF* model? This is precisely the goal of all studies trying to derive mean-field descriptions of *spiking neural networks*. This approach is promising but yet to be improved.

[I] cf. http://apt.cs.manchester.ac.uk/projects/SpiNNaker/project/
[II] cf. https://brainscales.kip.uni-heidelberg.de/

7.2 Project Global Impacts

7.2.1 Ethical Impact

The field of computational neuroscience has made enormous strides in recent years, with the scale and biological detail of neural simulations increasing regularly. Izhikevich and Edelman [49] performed, 13 years ago, a large-scale, biologically-detailed simulation of mammalian cortex containing one million neurons [49], based on global thalamocortical anatomy of a human brain with 10^{11} neurons.As we approach the point at which scientists will be able to work with biologically-detailed models, ethical implications of large-scale simulations of brains seems to hinge on the question of whether a computational simulation of a brain could have a phenomenal consciousness or to which extent. According to the Block [50]'s theory, to get access to the state they are in through sensory experiences.

The question of consciousness in computational systems has previously been considered in the context of artificial intelligence research [51]. In an hypothetical situation, an in silico neuron is capable of performing all the functions of a biological neuron. It would appear likely that we could eventually convert a living biological brain into a in silico brain without an interruption of consciousness: Individual neurons are generally held to contribute very little to conscious experience [52]. In and of itself, this does not demonstrate that consciousness can exist in computational systems, but rather only suggests that consciousness does not rely on biological neurons. However, what if, instead of an in silico neuron, a computational simulation of a neuron is used? As we expand the simulation, we would eventually reach a point where the entirety of the brain is comprised of interfaced computational neurons. It seems possible that the end result of such a process could be a conscious computational simulation of a brain. Of course, models or simulations with a generally high level of biological detail may not model the all the information processing properties of neurons. The above example and argument are not sufficient to demonstrate that it is possible for a computational simulation of a brain to possess some degree of consciousness, and therefore to potentially be capable of suffering. If, as we argue above, we must consider extending protections against suffering to simulations. We must also consider the "inverse" ethical question: Given uncertainty that a computational simulation is conscious, should we not use simulations instead of the corresponding biological organisms whenever possible?

7.2.2 Social Impact

Indeed, how should we judge a society whose citizens are not so much subjects but individuals who understand themselves and each other in neurological terms rather than as moral agents? But, whether or not specific neurotechnologies are designed to alter our social behaviour, their introduction and use will certainly have a social impact. In this light, a purely ethical assessment of potential neurotechnologies as the Human Brain Project social team aims to do (cf. Salles and Farisco [48]) seems critically incomplete. Any neurotechnology that purports to alter our subjective point of view is essentially political in nature [53]. For instance, debates about ketamine use for anxiety or to reproduce sleep state under in-vivo and in silico models [54] almost entirely ignore sociological research which have indicated that ketamine is being used in social rather than medical and scientific settings in many parts of the world [55] within a drug use environment which triggers schizophrenic symptoms on mentally-healthy patients or seizures in epileptic patients. This promotion changed the perception of such treatment under our drug policy. There needs to be a greater level of dialogue and engagement between neuroscience and social science if we are to use the knowledge and technologies that emerge from this domain in a politically responsible manner.

7.2.3 Environmental Impact

In fact, part of the work done in computational neuroscience aims at connecting biological mechanisms with preexisting computer algorithms (e.g. seeing synaptic plasticity as a biological implementation of independent component analysis) [56]. What is wrong nowadays is that, the entire field seems to take the computer as a model of the brain. Neural models that address cognitive abilities are generally developed under the input-output paradigm: feed data in (i.e in my project experimental physiological measurements), get results out (firing rates). The environment of neural models is often inspired by a specific controlled lab experiment as mentioned in subsection. 7.1.2. As a result, it is not so clear that they have any explanatory value for ecological situations or be considered as models of a biological organism like the *cortical columns* I tried to model here. Instead, the assessment of model performance must rely on the intervention of the external observer, as in the coding paradigm: models are designed so as to "encode" features in the world, meaning that the external observer, not the organism, decodes the activity of the model. This issue again comes from the computer analogy, in which the output of a program is meaningful only because an external observer gives it a meaning [57].

There is another criticism of the computer analogy, which has to do with the idea that the brain has been engineered by evolution to fulfill a function in the same way as a computer is engineered. So in virtually all neuron models, there are a number of parameters (for example time constants) whose values are either chosen "because it works" like the "tricks" mentioned in subsection 7.1.2, or because of indirect experimental measurements such as EEG signals. But genes do not encode parameter values, result from all these interactions, in the body and with the environment. The structure of the brain is highly dynamic. A different approach to computational neuroscience could take biological organisms, rather than the computer, as models. Neural models must be embedded in an environment and interact with it, so that the external observer is not part of the cognitive process. Moreover, organizational mechanisms that guarantee sustainability in an unknown environment should be focused on.

7.3 Retrospection

7.3.1 Difficulties encountered

The following is a description of the obstacles and difficulties encountered during the research process.However, the reality was quite different, some obstacles were difficult to overcome and some unexpected ones occurred during the course of the project.

A. **The Lack of Communication** has been inherent between the research team members with myself. The pandemic situation forced us to work remotely at the beginning. Afterwards, the sharing workflow was only made by mails or few zoom meetings. Every Monday, a brainstorm session was held for the whole laboratory in order to share each other thoughts on someone's research project.However, it was uncomfortable for me to share thoughts on subjects I was not familiar with and in front of postdocs or professors. Moreover, my supervisors let me work alone a lot but apparently in the research field the amount of autonomy for an intern is way larger than in a regular firm, which resulted in cuts in communication for several weeks.Since I started working on my own, no team member was aware of the overall progress of the project on a weekly basis. However, this form of management increased dramatically my work autonomy which helped me conducting a research project at such an early stage of my professional experience.

B. **The Definition of the Project** was my biggest brake and led to unexpected risks such as the feeling of three months of work wasted. This resulted of a misunderstanding between the main tutor and the co-supervisors who were planning the project steps backwards from the initial project outline, the professor had in mind.As a result, it had a big impact on the project's progress. However, I learned this issue was inevitable in a research project most of the time, on this particular issue Pr. *Villers* helped me a lot to overcome the frustration that bounds to happen in order to step forward even if the results are not what I expected them to be, because what is the point of researching something then?

C. **Choosing the Right Methodology** was one of the most difficult and confusing decisions for me, because I discovered late in the research process that the type of research will dictate the right research methodologies that should underpin the research and data collection methods to be used. Regardless of the methods or methodology adopted for the study, the data collection techniques employed must be suitable and capable of meeting the objectives of the study. Moreover, it is important that the technique used in collecting data is adequate enough to provide the information required to accomplish the overall goals of the study. Therefore, the methodology that I used came from the defining properly the research question and after getting the first results that guided me which was unusual compared to the project I worked on when we already knew what we were looking for.

D. **Dealing with my Data** was the final challenge. How to make sense of the data I have gathered ? After, collecting a very large amount of data, I had to connect my own research to the existing research in order to present them in a way that demonstrates how this research project adds to the body of the existing knowledge. On one hand, for the qualitative part of this research, which was quite important, looking for patterns and studying them with reliable technology as the Brian2 Simulator [34] was invaluable, I could not have understood precisely such complex behaviours without this platform. Moreover, for the accuracy sake, I had to discuss with my supervisors the biases in interpretation that may have played into my conclusions. On the other hand, for the quantitative study, I needed to address the biases of the individuals, here the one *cortical column AdEx* and *MF* models before generalizing the results to larger population such as the interconnected models.

7.3.2 Personal Appraisal

At the time of retrospection and personal appraisal, I would say that my internship was a the most complete and rewarding work experience in my life yet, which enabled putting into practice the knowledge acquired during my engineering formation in Applied Mathematics and Computer Science at Polytech-Sorbonne.

Initially, I picked this research laboratory for several reasons. First of all, the C.N.R.S. itself is a worldwide well-known institution in all the scientific field and the U.N.I.C. laboratory is rather renowned in the computational neuroscience field , working with international laboratories or other research institutions such as the Human Brain Project and even created The European Institute for Theoretical Neuroscience (EITN) funded by the European Commission itself. Furthermore, because of my medical background that I still have an interest in, I have planned to make a good use of my engineering formation in the biomedical field, to link the technology with the living experience.

One of the most positive aspect of my internship is in all likelihood the interdisciplinary aspect of my research project which had a dramatic self-improvement impact. This subject goes beyond my core speciality, therefore I was able to add useful skills and knowledge which locate at several majors intersections such as Computer Science and Biology (e.g. Brian2 simulator of neurons [34]) or Mathematics and Physics (e.g. Semi-analytical approach of the mean-field's transfer function theory). I even decided to go further by taking computational neuroscience online courses provided by the University of Washington. Thanks to this multidimensional topic, I have finally a grasp of what all the hard and soft proficiencies acquired through my studies could be used in the work environment by putting into practice methods or knowledge on unfamiliar fields' project while achieving the task or the research progress expected. Another aspect deep-rooted to the project interdisciplinary is the diversity of the research team members. Everyone was more or less an expert in a precise area of endeavour: Pr. Destexhe in Physical Chemistry and Theoretical Biology [3], Ph.D. Carlu in Applied Physics more precisely in Complex Systems and Mathematical Biology [5], Ph.D. Depannmaecker in Experimental Neurosciences [4] and myself (not an expert yet) in Applied Mathematics and Computer Science. Given this circumstance, the exchanges were rather tough, we all had to surpass ourselves due to the several points of contention on the same concept considered differently in each field (e.g. mean field theory, chaos theory, topology, etc.) which sprang up new hypothesis and more solid basis of notions I thought I had mastered or at least knew enough about it.

I had the opportunity to witness the relevancy of applying exact science for the benefits of experimental ones which led to more accuracy from Mathematics (e.g. Markov hypothesis and Fokker-Plank equation to the mean-field model) but in reverse some biases were incorporated in the discovered phenomena not proven as precisely as a theorem demonstration (e.g. the Markov hypothesis not valid anymore for the master equation at the second order).

Following, this remark, I experienced even before being a researcher myself, the frustration and isolation of researchers. The complexity of topic (which could conceivably be extended to a thesis) and the research field being relatively new (i.e. derived from the artificial intelligence in order to get closer to an in silico brain twin) bring out constant changes, with only few references, it made my work full of endless researches while being clueless about the expectations.

Assuming I would have to evaluate my work, I would honestly disclose I am half satisfied because I was not able to give suitable results for the interconnected *AdEx* even though this part became optional but "the cherry on top" in order to really assess the validity scope of the interconnected mean-field, if I had enough time. As well as other incomplete results or interpretation (relevant or not), they leave me on my end with a strong incomplete aftertaste. It motivates me to espy the possibility to deepen the subject by carrying out the above mentioned part and trying to implement the improvement suggestions that I have proposed earlier.

Eventually, this scientific and human exploit got me almost ready for leading research projects of larger range and this is a priceless ability.

A. Parameters tables

Ahead of digging into the subject core, here displayed the lists of the fundamental variables which characterized neurons, synapses and neural network. Most of the values come from the literature Destexhe [22], Górski, Depannemaecker, and Destexhe [29], Boustani and Destexhe [30], and Naud, Marcille, Clopath, *et al.* [58] to ensure a the most realistic model based on biological experiment measurements.

★ means these values changes during the configuration we chose for a simulation, ■ means these values are computed during the simulations.

Parameters	Symbols	Description	Value	Unit (SI)
Neudel	V_m	membrane potential	65.10^{3}	Volt V
	g_l	leak conductance	10.10^{-9}	Siemens S
	C_m	membrane capacitance	200.10^{-12}	Farad F
	τ_{refrac}	refractory period	5.10^{-3}	Seconds s
FS cells	E_l	leak reversal potential	-65.10^{-3}	Volt V
	E_i	*FS* cells reversal potential	-80.10^{-3}	Volt V
	G_{syn_i}	synaptic conductance	cf. equation 5.1	Second s
	Q_i	inhibitory synaptic weight (conductance increment)	5.10^{-9}	Second s
	D_t	slope factor	2.10^{-3}	Volt V
	T_{syn}	synaptic period	5.10^{-3}	Second s
	τ_w	adaptation time constant	1.10^{-3}	Seconds s
	V_t	threshold potential	-50.10^{-3}	Volt V
	w	adaptation variable	0	Volt V
	a	adaptation conductance	0	Siemens S
	b	adaptation conductance increment	0	Siemens S
RS cells	E_l	leak reversal potential	-70.10^{-3}	Volt V
	E_e	*RS* cells potential	0.10^{-3}	Volt V
	G_{syn_e}	synaptic conductance	cf. equation 5.1	Second s
	Q_e	excitatory synaptic weight (conductance increment)	★ $.10^{-9}$	Second s
	D_t	slope factor	5.10^{-4}	Volt V
	T_{syn}	synaptic period	5.10^{-3}	Second s
	w	adaptation variable	■ $.10^{-12}$	Volt V
	a	adaptation conductance	0.10^{-12}	Siemens S
	b	adaptation conductance increment	60.10^{-12} or 1.10^{-12} ★	Siemens S
	τ_w	adaptation time constant	1	Seconds s
	V_t	threshold potential	-50.10^{-3}	Volt V
System properties	I_{syn}	synaptic input	★$.10^{-9}$	Ampere A
	prbc	synaptic connectivity probability	5	percent %
	N	population size	$\frac{80}{20}$ for $\frac{FS}{RS}$	None
	P_{ed}	external drive (Poisson Group)	2.5 Hz or 0.8 ★	Hertz Hz

Table A.1: *Spiking neural networks* and mean-field models parameters

★ means the value will/can change depending on the configuration, ■ means the value is computed within the code.

Parameters	Value	Description
$N0_e$	8000	Assigned value for *FS* cells
$N0_i$	2000	Assigned value for *RS* cells
tstep	0.5 ms	time steps
tot_numb	★	numbers of variables : (2 for *RS-FS* model, 4 for interconnected models)
N	8000 ★ or 2000 ★	size of *RS* and *FS* in one population
numb_var	★	≡ numbers of population, (2 for *RS-FS* model 4 for interconnected models)
numb_adapt	0 for *FS* cells or 1 for *RS*	adaptation size
F	■	transfer function
h	0.0001	correction term
T	0.05 ms	time simulation
mu	■	mean
p_0	0.05	probability of synaptic connection
ext_drive	★	external input (Poisson or Ornstein–Uhlenbeck noise)
input_rate	★	rate of the input frequencies

Table A.2: MeanField_dN_dp function parameters

B. Relevant codes of Mean-field

B.1 Synapses and cell construction

B.1.1 Membrane equation based on Brain2

```
def get_membrane_equation(neuron_params, synaptic_array,\
                          return_equations=False):

    ## pure membrane equation
    if neuron_params['delta_v']==0:
        # if hard threshold : Integrate and Fire
        eqs = """
        dV/dt = (%(Gl)f*nS*(%(El)f*mV - V) + I - w_adapt)/(%(Cm)f*pF) : volt (unless refractory)
    """ % neuron_params
    else:
        eqs = """
        dV/dt = (%(Gl)f*nS*(%(El)f*mV*hetEl - V) + %(Gl)f*nS*%(delta_v)f*mV*exp(-(%(Vthre)f*mV-V)
    /(%(delta_v)f*mV)) + I - w_adapt+%(ampnoise)f*pA*xi*((%(Gl)f*nS)/(%(Cm)f*pF))**-0.5)/(%(Cm)f*
    pF) : volt (unless refractory)

             hetEl : 1


            """ % neuron_params

    ## Adaptation current
    if neuron_params['tauw']>0: # adaptation current or not ?
        eqs += """
        dw_adapt/dt = ( -%(a)f*nS*( %(El)f*mV - V) - w_adapt )/(%(tauw)f*ms) : amp

            """ % neuron_params
    else:
        eqs += """
        w_adapt : amp  """

    ##1-adding all synaptic currents to the membrane eq via the I variable
    eqs += """
        I = I0 """
    for synapse in synaptic_array:
        # loop over each presynaptic element onto this target
        Gsyn = 'G'+synapse['name']
        eqs += '+'+Gsyn+'*(%(Erev)f*mV - V)' % synapse
    eqs += ' : amp'

    ##2-constructing the temporal dynamics of the synaptic conductances
    for synapse in synaptic_array:
        # loop over each presynaptic element onto this target
        Gsyn = 'G'+synapse['name']
        eqs += """
        """+'d'+Gsyn+'/dt = -'+Gsyn+'*(1./(%(Tsyn)f*ms)) : siemens' % synapse
    eqs += """
        I0 : amp """
    # adexp, pratical detection threshold Vthre+5*delta_v
    neurons = brian2.NeuronGroup(neuron_params['N'], model=eqs,\
                                 refractory=str(neuron_params['Trefrac'])+'*ms',
                                 threshold='V>'+str(neuron_params['Vthre']+5.*neuron_params['
    delta_v'])+'*mV',
                                 reset='V='+str(neuron_params['Vreset'])+'*mV; w_adapt+='+str(
    neuron_params['b'])+'*pA')
    neurons.hetEl=np.random.normal(1., 0, neuron_params['N'])

    #print(eqs)
    if return_equations:
        return neurons, eqs
    else:
        return neurons
```

B.1.2 Cells Synapses and External drive connections

```
def build_up_recurrent_connections(NeuronGroups, syn_conn_matrix):

    CONNECTIONS = np.empty((len(NeuronGroups), len(NeuronGroups)))

    if len(NeuronGroups)!=syn_conn_matrix.shape[0]:
        print('problem the matrix of connectivity and synapses does not')
        print('match the number of populations')

    for jj in range(len(NeuronGroups)):
        # loop over post-synaptic neurons
        for ii in range(len(NeuronGroups)):
            # loop over presynaptic neurons
            CONNECTIONS[ii,jj] = Synapses(NeuronGroups[ii], NeuronGroups[jj],\
                                          model='w:siemens',\
                                          name=syn_conn_matrix[ii,jj]['name'],
                                          pre='G'+syn_conn_matrix[ii,jj]['name']+'_post +=w')
            CONNECTIONS[ii,jj].connect('i != j', p=syn_conn_matrix[ii,jj]['p_conn'])  # exclude
    self-connections
            CONNECTIONS[ii,jj].w = syn_conn_matrix[ii,jj]['Q']*nS
    return CONNECTIONS

def build_up_poisson_group_to_pop(list_of_freqs, list_of_synapses, target_group):

    CONNECTIONS = np.empty(len(NeuronGroups), dtype=object)

    return CONNECTIONS
```

B.2 Transfer function

B.2.1 TF parameters and useful function

```
def pseq_params(params):
    Qe, Te, Ee = params['Qe'], params['Te'], params['Ee']
    Qi, Ti, Ei = params['Qi'], params['Ti'], params['Ei']
    Gl, Cm , El = params['Gl'], params['Cm'] , params['El']
    for key, dval in zip(['Ntot', 'pconnec', 'gei'], [1, 2., 0.5]):
        if key in params.keys():exec(key+' = params[key]')

        else:exec(key+' = dval')
    if 'P' in params.keys():P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, P10 = params['P']
    else:P0 = -45e-3
       for i in range(1,11):
           exec('P'+str(i)+'= 0')

    if 'P' in params.keys(): return Qe,Te, Ee, Qi, Ti, Ei, Gl, Cm, El, params['Ntot'], params['
    pconnec'], params['gei'], P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, P10
    else : return Qe,Te, Ee, Qi, Ti, Ei, Gl, Cm, El, params['Ntot'], params['pconnec'], params['
    gei'], P0


def get_fluct_regime_varsup(Fe, Fi, XX,Qe, Te, Ee, Qi, Ti, Ei, Gl, Cm, El, Ntot, pconnec, gei, P0
    ):
    fe = Fe*(1.-gei)*pconnec*Ntot # default is 1 !!
    fi = Fi*gei*pconnec*Ntot
    muGe, muGi = Qe*Te*fe, Qi*Ti*fi
    muG = Gl+muGe+muGi
    muV = (muGe*Ee+muGi*Ei+Gl*El-XX)/muG
    muGn, Tm = muG/Gl, Cm/muG
    Ue, Ui = Qe/muG*(Ee-muV), Qi/muG*(Ei-muV)
    sV = np.sqrt(fe*(Ue*Te)**2/2./(Te+Tm)+fi*(Ti*Ui)**2/2./(Ti+Tm))
    fe, fi = fe+1e-9, fi+1e-9 # just to insure a non zero division,
    Tv = ( fe*(Ue*Te)**2 + fi*(Ti*Ui)**2 ) /( fe*(Ue*Te)**2/(Te+Tm) + fi*(Ti*Ui)**2/(Ti+Tm) )
    TvN = Tv*Gl/Cm

    return muV, sV+1e-12, muGn, TvN


def mean_and_var_conductance(Fe, Fi, Qe, Te, Ee, Qi, Ti, Ei, Gl, Cm, El, Ntot, pconnec, gei, P0):

    fe = Fe*(1.-gei)*pconnec*Ntot # default is 1 !!
    fi = Fi*gei*pconnec*Ntot
    return Qe*Te*fe, Qi*Ti*fi, Qe*np.sqrt(Te*fe/2.), Qi*np.sqrt(Ti*fi/2.)

### FUNCTION, INVERSE FUNCTION
def erfc_func(muV, sV, TvN, Vthre, Gl, Cm):
    return .5/TvN*Gl/Cm*(sp_spec.erfc((Vthre-muV)/np.sqrt(2)/sV)))

def effective_Vthre(Y, muV, sV, TvN, Gl, Cm):
    Vthre_eff = muV+np.sqrt(2)*sV*sp_spec.erfcinv(\
                    Y*2.*TvN*Cm/Gl) # effective threshold
    return Vthre_eff

def threshold_func(muV, sV, TvN, muGn, P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, P10):

    muV0, DmuV0 = -60e-3,10e-3
    sV0, DsV0 =4e-3, 6e-3
    TvN0, DTvN0 = 0.5, 1.

    return P0+P1*(muV-muV0)/DmuV0+\
        P2*(sV-sV0)/DsV0+P3*(TvN-TvN0)/DTvN0+\
        0*P4*np.log(muGn)+P5*((muV-muV0)/DmuV0)**2+\
        P6*((sV-sV0)/DsV0)**2+P7*((TvN-TvN0)/DTvN0)**2+\
        P8*(muV-muV0)/DmuV0*(sV-sV0)/DsV0+\
        P9*(muV-muV0)/DmuV0*(TvN-TvN0)/DTvN0+\
        P10*(sV-sV0)/DsV0*(TvN-TvN0)/DTvN0
```

B.2.2 TF final template

```
def TF_my_templateup(fe, fi,XX, Qe, Te, Ee, Qi, Ti, Ei, Gl, Cm, El, Ntot, pconnec, gei, P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, P10):

    if(hasattr(fe, "__len__")):fe[fe<1e-8]=1e-8

     else:

        if(fe<1e-8):fe=1e-8

    if(hasattr(fi, "__len__")):fi[fi<1e-8]=1e-8

    else:

        if(fi<1e-8):fi=1e-8

    muV, sV, muGn, TvN = get_fluct_regime_varsup(fe, fi,XX, Qe, Te, Ee, Qi, Ti, Ei, Gl, Cm, El, Ntot, pconnec, gei, P0)#, P1, P2, P3, P4, P5, P6, P7, P8, P9, P10)
    Vthre = threshold_func(muV, sV, TvN, muGn, P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, P10)

    if(hasattr(muV, "__len__")):sV[sV<1e-4]=1e-4

    else:

        if(sV<1e-4):sV=1e-4

    Fout_th = erfc_func(muV, sV, TvN, Vthre, Gl, Cm)
    if(hasattr(Fout_th, "__len__")):Fout_th[Fout_th<1e-8]=1e-8

    else:

        if(Fout_th<1e-8):Fout_th=1e-8

    return Fout_th

def gaussian(x, mu, sig):
    return (1/(sig*np.sqrt(2*3.1415)))*np.exp(-np.power(x - mu, 2.) / (2 * np.power(sig, 2.)))

def make_loop(t, nu, vm, nu_aff_exc, nu_aff_inh, BIN,\
              Qe, Te, Ee, Qi, Ti, Ei, Gl, Cm, El, Ntot, pconnec, gei, P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, P10):
    dt = t[1]-t[0]
    # constructing the Euler method for the activity rate
    for i_t in range(len(t)-1): # loop over time

        fe = (nu_aff_exc[i_t]+nu[i_t]+Fdrive) #afferent+recurrent excitation
        fi = nu[i_t]+nu_aff_inh[i_t] #recurrent inhibition
       W[i_t+1] = W[i_t] + dt/Tw*(b*nu[i_t]*Tw - W[i_t])
        nu[i_t+1] = nu[i_t] +\
               dt/BIN*(\
                TF_my_template(fe, fi, W[i_t], Qe, Te, Ee, Qi, Ti, Ei, Gl, Cm, El, Ntot, pconnec, gei, P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, P10)\
                -nu[i_t])
        vm[i_t], _, _, _ = get_fluct_regime_vars(fe, fi, W[i_t], Qe, Te, Ee, Qi, Ti, Ei, Gl, Cm, El, Ntot, pconnec, gei, P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, P10)

    return nu, vm, W
```

B.2.3 Fitting data to TF

```
def make_fit_from_data(DATA, with_square_terms=False):
    for r_ps, d_ps, f_ps in os.walk(DATA):
        f_ps=f_ps

    print(DATA)

    MEANfreq=[]
    SDfreq=[]
    w=[]
    fe=[]
    fi=[]

    for fname in f_ps:

        filename=DATA+'/'+fname

        fe_loc,fi_loc,MEANfreq_loc, SDfreq_loc,\
        params,w_loc = np.load(filename,allow_pickle=True,encoding="latin1")
        fe.append(fe_loc)
        fi.append(fi_loc)
        w.append(w_loc)
        SDfreq.append(SDfreq_loc)
        MEANfreq.append(MEANfreq_loc)

    MEANfreq=np.array(MEANfreq)
    SDfreq=np.array(SDfreq)
    w=np.array(w)
    fe=np.array(fe)
    fi=np.array(fi)

    print(params)
    print(params.keys())

    P = fitting_Vthre_then_Fout(MEANfreq, fe, fi,w, params,\
                                with_square_terms=with_square_terms)

    print (1e3*np.array(P), 'mV')

    # then we save it:
    filename = DATA+ '_fit.npy'
    print ('coefficients saved in ', filename)
    np.save(filename, np.array(P))

    return P
```

B.2.4 Loading the TF

```
def load_transfer_functions_with_ALL_params_ij(NRN1, NRN2, NTWK, par, i, j):
#ij refers to the indices of the population for multi MF
    # NTWK
    M = get_connectivity_and_synapses_matrix(NTWK, SI_units=True)


    M[0,0]['Q'],M[0,1]['Q']=par['qe'],par['qe']
    M[1,0]['Q'],M[1,1]['Q']=par['qi'],par['qi']

    M[0,0]['Tsyn'],M[0,1]['Tsyn']=par['tau_e'],par['tau_e']
    M[1,0]['Tsyn'],M[1,1]['Tsyn']=par['tau_i'],par['tau_i']

    # NRN1
    params1 = get_neuron_params(NRN1, SI_units=True)

    if (par['diff'][i]==True):

        params1['El']=par['El'][i]
        params1['Cm']=par['Cm'][i]
        params1['Gl']=par['Gl'][i]
        params1['Vthre']=par['Vthre'][i]
        params1['b']=par['b'][i]
        params1['a']=par['a'][i]
        params1['tauw']=par['tauw'][i]# here divide by 1000 because tauw is in seconds in MF
    codes

    reformat_syn_parameters(params1, M)

    try:

        P1 = np.load('transfer_functions/data/'+NRN1+'_'+NTWK+'_fit.npy')

        params1['P'] = P1
        def TF1(fe, fi,XX):
            return TF_my_templateup(fe, fi,XX, *pseq_params(params1))

    except IOError:
        print('=====  fit for NRN1 not available ==========')

    # NRN1
    params2 = get_neuron_params(NRN2, SI_units=True)

    if (par['diff'][j]==True):

        params2['El']=par['El'][j]
        params2['Cm']=par['Cm'][j]
        params2['Gl']=par['Gl'][j]
        params2['Vthre']=par['Vthre'][j]
        params2['b']=par['b'][j]
        params2['a']=par['a'][j]
        params2['tauw']=par['tauw'][j]


    reformat_syn_parameters(params2, M)

    try:

        P2 = np.load('transfer_functions/data/'+NRN2+'_'+NTWK+'_fit.npy')

        params2['P'] = P2
        def TF2(fe, fi,XX):
            return TF_my_templateup(fe, fi,XX, *pseq_params(params2))

    except IOError:
        print('=====  fit for NRN2 not available =========')

    return TF1, TF2
```

B.3 Orstein-Uhlenbeck noise

```
def Ornstein_U_list(xi_0,dt,tot_time,tau):

    for i in np.arange(0,tot_time,dt):
        xi_0+=-dt*xi_0/tau+np.sqrt(dt)*np.random.randn()

###   To integrate in the MF, it needs to be written as a function

def OU_func_list(t,O_U_list):
    return O_U_list[t]

def Ornstein_func(xi_0,dt,tau):

    return xi_0-dt*xi_0/tau+np.sqrt(dt)*np.random.randn()
```

Integrated as a parameter in the numerical integration :

```
np.random.seed(10)
import datetime ### To keep track of time
x0=np.ones(params['numb_var'])*1e-5
x0[0]=0.6 #ve1 exp
x0[1]=0.6 #ve2 exp
x0[2]=7.0 #vi1 exp
x0[3]=7.0 #vi2 exp
x0[20]=x0[0]*1e-12 #we1
x0[21]=x0[1]*1e-12 #we2
print(x0)
time=0
t_trace2=[]
x_trace_RS_with_noise=[]
TF_trace_RS_with_noise=[]

drive=[]
params['tstep']=1e-4
params['T']=0.005


xi_exc=0 ## O_U process
xi_inh=0


print(x0)
time=0

for i in np.arange(0,10,params['tstep']):
    TF_inst= Euler_method_General(x0,params,time,MeanField_dN_dp_Hybrid_noise)

    xi_exc=Ornstein_func(xi_exc,params['tstep'],5e-4)
    xi_inh=Ornstein_func(xi_inh,params['tstep'],5e-4)

    params['ext_inp'][0]=2.0+10.5*xi_exc
    params['ext_inp'][1]=2.0+10.5*xi_inh

    if (i%0.5==0):
        print('i = ', i, '\n')
        print('time : ', datetime.datetime.now(), ' \n \n')
    time+=params['tstep']
    t_trace2.append(time)

    x_trace_RS_with_noise.append(x0.copy())
    TF_trace_RS_with_noise.append(TF_inst.copy())
    drive.append(params['ext_inp'])
```

C. Codes of Network Quantification for AdEx

```
fig, axs = plt.subplots(2, 1, figsize=(4, 3))
sp = M1G2.i
times = M1G2.t/ms
nNeurons = max(sp) + 1
nSpikes = len(sp)

spiketimes = {}
for n in range(nNeurons):
    ts = list(times[(np.where(sp == n))])
    #ts = [t for t in ts if t > 500 * ms]
    spiketimes[n] = ts
isi_stds = []
isi_cvs = []
for n in range(nNeurons):
    isi_std = np.std(np.diff(spiketimes[n]))
    isi_cv = np.std(np.diff(spiketimes[n])) / np.mean(np.diff(spiketimes[n]))
    isi_stds.append(isi_std)
    isi_cvs.append(isi_cv)

meanFiringRates = []
for n in range(nNeurons):
    meanFiringRates.append(np.mean(1/(np.diff(spiketimes[n]))))



axs[0].hist(isi_cvs, bins=15, alpha=0.9, edgecolor='black');
axs[1].hist(meanFiringRates,bins=15, alpha=0.9, edgecolor='black',color="green");

for k in range(1):
    leg = axs[k].legend(loc=1)
    leg.get_frame().set_facecolor('none')
    leg.get_frame().set_linewidth(0.0)

for k in range(2):
    axs[k].spines['right'].set_visible(False)
    axs[k].spines['top'].set_visible(False)
    axs[k].spines['bottom'].set_visible(False)
    axs[k].spines['left'].set_visible(False)

axs[0].set_xlabel("CV$_{ISI}$")
axs[1].set_xlabel("CC")

axs[0].set_ylabel("Neuron count")
axs[1].set_ylabel("Neuron count")

plt.subplots_adjust(hspace=0.5)
```

D. Management Tools

From the start of the project and for several months, we set out the important steps to be carried out in the form of a *WBS* and *Gantt* chart on the Fig. D.1.

However, as our progress has not been so orderly, I have changed my methodology thanks to Mrs.*Villers* experience in research field/project.

I have learned at my internship midterm that by its very nature, completing an independent research project on time, and to the best of your abilities, requires the completion of multiple, interrelated activities, and multiple deadlines. You will need to be able to respond dynamically to any additional challenges you might face along the way. And, at the same time, your research project might well be just one of a number of key activities that you are involved in at that time.

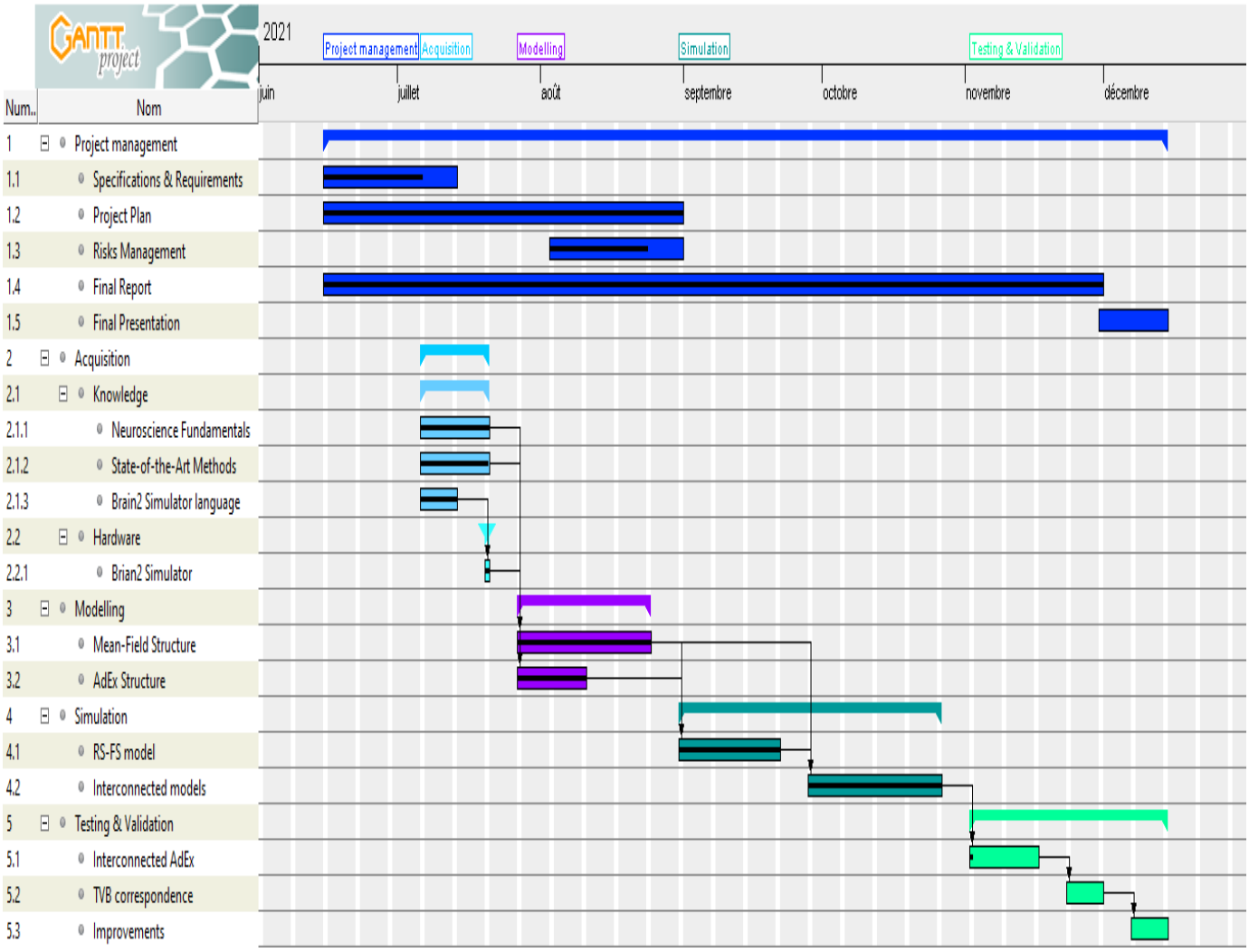


Figure D.1: Gantt chart of the Research Project

Glossary

Artificial intelligence Artificial intelligence is the study of systems enable to correctly interpret external data, draw lessons from that data, and use those lessons to achieve specific objectives and tasks.
✍ Page 33.

Artificial neural network Artificial neural network is a computational model that consists of several processing elements that receive inputs and deliver outputs based on their predefined activation functions..
✍ Page 1.

Attractor In the mathematical field of dynamical systems, an attractor is a set of states toward which a system tends to evolve, for a wide variety of starting conditions of the system. System values that get close enough to the attractor values remain close even if slightly disturbed..
✍ Pages V, 2, 25, 26, 29.

Bifurcation Small changes of the parameters of a nonlinear system can cause sudden and dramatic changes of its overall behavior..
✍ Pages V, 1, 23, 26, 28–30.

Cerebral cortex The cerebral cortex is an organic tissue, also called grey matter, which covers the two hemispheres of the brain to a thickness of a few millimetres. It plays a crucial role in relation to the basic nervous functions: these are the different modalities that are motor, sensitivity, sensory..
✍ Pages 1, 4, 5, 19, 24, 29, 30.

Connectome A connectome is a comprehensive map of neural connections in the brain, and may be thought of as its "wiring diagram". A connectome is constructed by tracing the neuron in a nervous system and mapping where neurons are connected through synapses..
✍ Pages 2, 30.

Cortical column A cortical column is a group of neurons located in the cerebral cortex that have identical reception fields. Cortical columns are sometimes considered to be units of the cortex computations. They have proximity connectivity, with neighbouring columns, and long-distance connectivity, from area to area or hemisphere to hemisphere, via myelinated axons..
☞ See cerebral cortex.
✍ Pages 1, 4, 5, 7, 9, 10, 13, 19, 20, 22, 23, 29–32.

Depolarization Membrane depolarization is a decrease in the difference in potential between the extracellular and the intracellular medium on either side of the plasma membrane, most often due to the intracellular entry of positive ions..
☞ See spike.
✍ Pages 3, 5.

Fast-spiking Subtype of inhibitory neocortical interneuron.
☞ See spiking neuron model.
✍ Pages 1, 3–5.

Fokker-Plank equation In statistical mechanics, the Fokker–Planck equation is a partial differential equation that describes the time evolution of the probability density function of the velocity of a particle under the influence of drag forces and random forces, as in Brownian motion..
✍ Pages 9, 33.

Homoclonic orbit A homoclinic orbit is a trajectory of a flow of a dynamical system which joins a saddle equilibrium point to itself. More precisely, a homoclinic orbit lies in the intersection of the stable manifold and the unstable manifold of an equilibrium..
✍ Page 26.

Hyperpolarization Hyperpolarization is a change in a cell's membrane potential that makes it more negative. It is the opposite of a depolarization..
☞ See spike.
✍ Pages 3, 5.

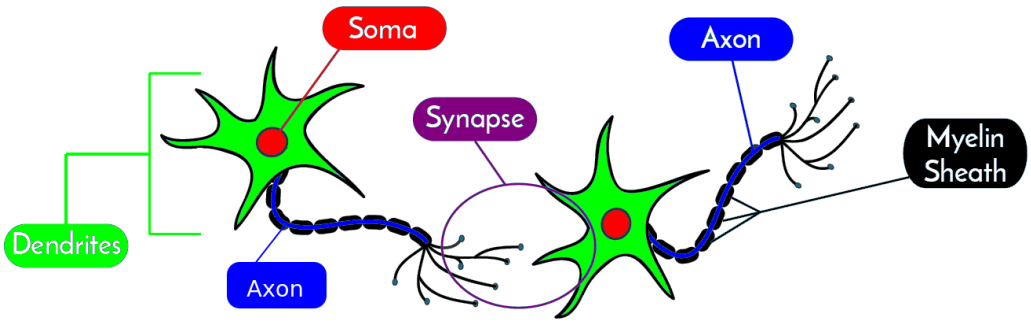
In silico In biology and other experimental sciences, an in silico experiment is one performed on computer or via computer simulation..
✍ Pages 1, 31, 33.

Limit cycle A limit cycle is a closed trajectory in phase space having the property that at least one other trajectory spirals into it either as time approaches infinity or as time approaches negative infinity.
✍ Pages V, 23–26, 28, 29.

Machine learning The scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying instead on models and inferences. This discipline is considered a subset of the artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, called "training data", to make predictions or decisions without being explicitly programmed to perform the task.
✍ Pages 1, 2.

Neural network Biological neural network.
✍ Pages 1, 2, 5, 6, 9, 10, 12, 30, 34.

Neuron A neuron or nerve cell is an electrically excitable cell that communicates with other cells via specialized connections called synapses. A typical neuron consists of a cell body known as soma, dendrites, and a single axon covered by myelin sheath as displayed.



✍ Pages V, 1–13, 17, 24, 26, 27, 30, 31, 33, 34.

Neurotransmitter Neurotransmitters are chemical messengers that transmit a signal from a neuron across the synapse to a target cell, which may be another neuron, a muscle cell, or a gland cell. Neurotransmitters are chemical substances made by the neuron specifically to transmit a message.
✍ Page 3.

Ornstein–Uhlenbeck The Ornstein–Uhlenbeck process is a stationary Gauss–Markov process, which means that it is a Gaussian process, a Markov process, and is temporally homogeneous. The process can be considered to be a modification of the random walk in continuous time, or Wiener process.

$$dx_t = -\theta x_t dt + \sigma dW_t \tag{D.1}$$

The Wiener process with ξ a random standard normal variable, for t $\in$ [0,1]

$$W_n(t) = \frac{1}{\sqrt{n}} \sum_{1 \leqslant k \leqslant \lfloor nt \rfloor} \xi_k \tag{D.2}$$

☞ See Fokker-Plank equation.
✍ Pages 5, 18, 30.

Phase plane Consider a systems differential equations y" = F(y,y'). In this context, the Cartesian plane where the phase portrait resides is called the phase plane. The parametric curves traced by the solutions are sometimes also called their trajectories. The phase plane method refers to graphically determining the existence of limit cycles in the solutions of the differential equation..
☞ See limit cycle.
✍ Pages 22, 24–26.

Place cell A place cell is a kind of pyramidal neuron within the hippocampus that becomes active when an animal enters a particular place in its environment, which is known as the place field. Place cells are thought, collectively, to act as a cognitive representation of a specific location in space, known as a cognitive map.
✍ Page 29.

Recurrent neural network A recurrent neural network is a class of artificial neural network where connections between nodes form a directed or undirected graph along a temporal sequence. This allows it to exhibit temporal dynamic behavior. The term "recurrent neural network" is used to refer to the class of networks with an infinite impulse response..
☞ See artificial neural network.
✍ Page 2.

Regular-spiking Excitatory neocortical interneuron.
☞ See spiking neuron model.
✍ Pages 1, 3–5.

Repolarization Repolarization refers to the change in membrane potential that returns it to a negative value just after the depolarization phase of an action potential which has changed the membrane potential to a positive value. The repolarization phase usually returns the membrane potential back to the resting membrane potential..
☞ See spike.
✍ Page 3.

Sparse plural Sparse coding consists of a unit of information which is represented by the activation of a small number of neurons, from a few neurons to a few tens of thousands of neurons at most. Sparse network has much fewer links than the possible maximum number of links within that network. .
✍ Page 1.

Spike Action potential.
✍ Pages 1, 3–5, 7–12, 14, 30.

Spiking neural network Spiking neural networks are artificial neural networks that more closely mimic natural neural networks. In addition to neuronal and synaptic state, they incorporate the concept of time into their operating model..
☞ See spiking neuron model.
✍ Pages VI, 1, 2, 6, 29, 30, 34.

Stochastic differential equation A stochastic differential equation is a differential equation in which one or more of the terms is a stochastic process, resulting in a solution which is also a stochastic process..
✍ Pages 5, 10, 11.

Transient Is a short-lived burst of energy in a system caused by a sudden change of state. It is a momentary event preceding the steady state..
✍ Pages 3, 22, 29.

Acronyms

AdEx Spiking neural network model.
✍ Pages 2, 5–8, 10, 18, 29, 30, 32, 33.

ANN *Artificial neural network*.
☞ See artificial neural network.
✍ Page 1.

C.N.R.S. Centre National de la Recherche Scientifique or *French National Centre for Scientific Research* (https://www.cnrs.fr/).
✍ Pages 2, 33.

FS *Fast-spiking*.
☞ See *fast-spiking*.
✍ Pages V, 1, 4, 8, 10, 16–18, 34, 35.

MF Studies the behavior of high-dimensional random (stochastic) models such as AdEx by studying a simpler model that approximates the original by averaging over degrees of freedom..
✍ Pages 5, 19, 20, 29, 30, 32.

NN *Neural network*.
☞ See neural network.
✍ Page 1.

RS *Regular-spiking*.
☞ See *regular-spiking*.
✍ Pages V, 1, 4, 8, 10, 16–18, 34, 35.

SDE *Stochastic differential equation*.
☞ See stochastic differential equation.
✍ Page 5.

SNN *Spiking neural network*.
☞ See *spiking neural network*.
✍ Pages 1, 29.

U.N.I.C. Unité de Neurosciences, Information et Complexité or *Unit of Neuroscience Information and Complexity* (https://neuropsi.cnrs.fr/fr/icn/alain-destexhe).
✍ Pages 2, 33.

References

[1] M. Pfeiffer and T. Pfeil, "Deep learning with spiking neurons: Opportunities and challenges," *Frontiers in Neuroscience*, vol. 12, p. 774, 2018, ISSN: 1662-453X. DOI: 10.3389/fnins.2018.00774. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnins.2018.00774.

[2] E. Daucé, "Apprentissage et Contrôle dans les Architectures Neuronales," Habilitation à diriger des recherches, Aix-Marseille Universite, Jan. 2016. [Online]. Available: https://tel.archives-ouvertes.fr/tel-01264905.

[3] A. Destexhe, "Nonlinear dynamics of the rhythmical activity of the brain," Ph.D. dissertation, Université libre de Bruxelles, Mar. 1992. [Online]. Available: https://tel.archives-ouvertes.fr/tel-01264905.

[4] D. T. Depannemaecker, "Redes neuronais realisticas e aprendizado," Ph.D. dissertation, Universidade Federal De Sao Paulo, 2019. [Online]. Available: https://sucupira.capes.gov.br/sucupira/public/consultas/coleta/trabalhoConclusao/viewTrabalhoConclusao.jsf?popup=true&id_trabalho=8001517.

[5] M. Carlu, "Instability in high-dimensional chaotic systems," Ph.D. dissertation, University of Aberdeen, 2019. [Online]. Available: https://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.76735.

[6] P. Vadapalli, *Spiking neural network: Everything you need to know*, https://www.upgrad.com/blog/spiking-neural-network, web article, 2020.

[7] M. Goldman, J. Raymond, and E. Aksay, *Plasticity of global brain dynamics: Tunable neural integration*, https://www.simonsfoundation.org/funded-project/plasticity-of-global-brain-dynamics-tunable-neural-integration/, research project, 2021.

[8] V. Mante, D. Sussillo, K. v. Shenoy, and W. T. Newsome, "Context-dependent computation by recurrent dynamics in prefrontal cortex," *Nature*, vol. 503, no. 7474, pp. 78–84, Nov. 2013. DOI: 10.1038/nature12742. [Online]. Available: https://www.nature.com/articles/nature12742.

[9] P. Dayan and L. Abbott, *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. Jan. 2001, vol. 15.

[10] V. Mountcastle, "Special issue: Computation in cortical columns: Introduction," *Cerebral cortex (New York, N.Y. : 1991)*, vol. 13, pp. 2–4, Feb. 2003. DOI: 10.1093/cercor/13.1.2.

[11] S. Chauvette, M. Volgushev, and I. Timofeev, "Origin of Active States in Local Neocortical Networks during Slow Sleep Oscillation," *Cerebral Cortex*, vol. 20, no. 11, pp. 2660–2674, Mar. 2010, ISSN: 1047-3211. DOI: 10.1093/cercor/bhq009. eprint: https://academic.oup.com/cercor/article-pdf/20/11/2660/17302400/bhq009.pdf. [Online]. Available: https://doi.org/10.1093/cercor/bhq009.

[12] J. S. Goldman, L. Kusch, B. H. Yalcinkaya, D. Depannemaecker, T.-A. E. Nghiem, V. Jirsa, and A. Destexhe, "Brain-scale emergence of slow-wave synchrony and highly responsive asynchronous states based on biologically realistic population models simulated in the virtual brain," *bioRxiv*, 2020. DOI: 10.1101/2020.12.28.424574. eprint: https://www.biorxiv.org/content/early/2020/12/29/2020.12.28.424574.full.pdf. [Online]. Available: https://www.biorxiv.org/content/early/2020/12/29/2020.12.28.424574.

[13] A. Rocha, "Toward a comprehensive understanding of eeg and its analyses," *SSRN Electronic Journal*, Jan. 2018. DOI: 10.2139/ssrn.3098803.

[14] M. Pfeiffer and T. Pfeil, "Deep learning with spiking neurons: Opportunities and challenges," *Frontiers in Neuroscience*, vol. 12, p. 774, 2018, ISSN: 1662-453X. DOI: 10.3389/fnins.2018.00774. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnins.2018.00774.

[15] M. L. L. Lapicque, "Recherches quantitatives sur l'excitation électrique des nerfs traitée comme une polarisation," *J. Physiol. Pathol.*, vol. 9, pp. 620–635, 1907.

[16] J. Schücker, C. Honerkamp, and M. Helias, "Mean-field theory for complex neuronal networks," Ph.D. dissertation, Universitätsbibliothek der RWTH Aachen, 2017.

[17] M. G. F. Fuortes and F. Mantegazzini, "Interpretation of the Repetitive Firing of Nerve Cells," *Journal of General Physiology*, vol. 45, no. 6, pp. 1163–1179, Jul. 1962, ISSN: 0022-1295. DOI: 10.1085/jgp.45.6.1163. eprint: https://rupress.org/jgp/article-pdf/45/6/1163/1242767/1163.pdf. [Online]. Available: https://doi.org/10.1085/jgp.45.6.1163.

[18] R. Brette and W. Gerstner, "Adaptive exponential integrate-and-fire model as an effective description of neuronal activity," *Journal of neurophysiology*, vol. 94, pp. 3637–42, Dec. 2005. DOI: 10.1152/jn.00686.2005.

[19] N. Fourcaud-Trocmé, D. Hansel, C. van Vreeswijk, and N. Brunel, "How spike generation mechanisms determine the neuronal response to fluctuating inputs," *The Journal of neuroscience : the official journal of the Society for Neuroscience*, vol. 23, pp. 11 628–40, Jan. 2004. DOI: 10.1523/JNEUROSCI.23-37-11628.2003.

[20] E. Izhikevich, *Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting*, ser. Computational neuroscience Dynamical systems in neuroscience. MIT Press, 2010, ISBN: 9780262514200. [Online]. Available: https://books.google.fr/books?id=RLdIPgAACAAJ.

[21] Y. Zerlaut, S. Chemla, F. Chavane, and A. Destexhe, "Modeling mesoscopic cortical dynamics using a mean-field model of conductance-based networks of adaptive exponential integrate-and-fire neurons," *Journal of Computational Neuroscience*, vol. 44, no. 1, pp. 45–61, Feb. 2018. DOI: 10.1007/s10827-017-0668-2. [Online]. Available: https://hal-amu.archives-ouvertes.fr/hal-02008319.

[22] A. Destexhe, "Self-sustained asynchronous irregular states and Up-Down states in thalamic, cortical and thalamocortical networks of nonlinear integrate-and-fire neurons.," *Journal of Computational Neuroscience*, vol. 27, no. 3, pp. 493–506, Dec. 2009. DOI: 10.1007/s10827-009-0164-4. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00444902.

[23] T. Kristensen and D. MacNearney, "Simulating spiking neurons by hodgkin huxley model."

[24] M. Breakspear and V. K. Jirsa, "Neuronal dynamics and brain connectivity," in *Handbook of Brain Connectivity*, V. K. Jirsa and A. McIntosh, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 3–64, ISBN: 978-3-540-71512-2. DOI: 10.1007/978-3-540-71512-2_1. [Online]. Available: https://doi.org/10.1007/978-3-540-71512-2_1.

[25] M. d. Volo, A. Romagnoni, C. Capone, and A. Destexhe, "Biologically realistic mean-field models of conductance based networks of spiking neurons with adaptation," *bioRxiv*, 2018. DOI: 10.1101/352393. eprint: https://www.biorxiv.org/content/early/2018/08/25/352393.full.pdf. [Online]. Available: https://www.biorxiv.org/content/early/2018/08/25/352393.

[26] N. Brunel and S. Sergi, *Firing frequency of leaky integrate-and-fire neurons with synaptic current dynamics*, 1998.

[27] C. Assisi, V. Jirsa, and S. Kelso, "Synchrony and clustering in heterogeneous networks with global coupling and parameter dispersion," *Physical review letters*, vol. 94, p. 018 106, Feb. 2005. DOI: 10.1103/PhysRevLett.94.018106.

[28] R. A. Stefanescu and V. Jirsa, "Reduced representations of heterogeneous mixed neural networks with synaptic coupling.," *Physical review. E, Statistical, nonlinear, and soft matter physics*, vol. 83 2 Pt 2, p. 026 204, 2011.

[29] T. Górski, D. Depannemaecker, and A. Destexhe, "Conductance-based adaptive exponential integrate-and-fire model," *bioRxiv*, 2020. DOI: 10.1101/842823. eprint: https://www.biorxiv.org/content/early/2020/08/20/842823.full.pdf. [Online]. Available: https://www.biorxiv.org/content/early/2020/08/20/842823.

[30] S. E. Boustani and A. Destexhe, "A Master Equation Formalism for Macroscopic Modeling of Asynchronous Irregular Activity States.," *Neural Computation*, vol. 21, no. 1, pp. 46–100, Jan. 2009. DOI: 10.1162/neco.2008.02-08-710. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00377137.

[31] H. Soula and C. Chow, "Stochastic dynamics of a finite-size spiking neural network," *Neural computation*, vol. 19, pp. 3262–92, Jan. 2008. DOI: 10.1162/neco.2007.19.12.3262.

[32] M. Massimini, F. Ferrarelli, R. Huber, S. K. Esser, H. Singh, and G. Tononi, "Breakdown of cortical effective connectivity during sleep," *Science*, vol. 309, no. 5744, pp. 2228–2232, 2005. DOI: 10.1126/science.1117256. eprint: https://www.science.org/doi/pdf/10.1126/science.1117256. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.1117256.

[33] S. Chevillard, "The functions erf and erfc computed with arbitrary precision and explicit error bounds," *Information and Computation*, vol. 216, pp. 72–95, 2012, Special Issue: 8th Conference on Real Numbers and Computers, ISSN: 0890-5401. DOI: https://doi.org/10.1016/j.ic.2011.09.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0890540112000697.

[34] D. Goodman and M. Stimber, *Brian2*, https://brian2.readthedocs.io/, 2020.

[35] S. Knapp, "Refractory period," *Biology Dictionary, Biologydictionary.net*, Nov. 2020. [Online]. Available: https://biologydictionary.net/refractory-period/.

[36] H. Zhang, J. Kang, T. Huang, X. Cong, S. Ma, and H. Huang, "Hopf bifurcation, hopf-hopf bifurcation, and period-doubling bifurcation in a four-species food web," *Mathematical Problems in Engineering*, vol. 2018, pp. 1–21, Sep. 2018. DOI: 10.1155/2018/8394651.

[37] J.-M. Ginoux, B. Muthuswamy, R. Meucci, S. Euzzor, A. Di Garbo, and K. Ganesan, "A physical memristor based muthuswamy–chua–ginoux system," *Scientific Reports*, vol. 10, no. 1, Nov. 2020, ISSN: 2045-2322. DOI: 10.1038/s41598-020-76108-z. [Online]. Available: http://dx.doi.org/10.1038/s41598-020-76108-z.

[38] R. Huzak, "Slow divergence integral on a möbius band," *Journal of Differential Equations*, vol. 266, no. 10, pp. 6179–6203, 2019, ISSN: 0022-0396. DOI: https://doi.org/10.1016/j.jde.2018.11.002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S002203961830648X.

[39] S. Ghosh, A. Mondal, P. Ji, A. Mishra, S. K. Dana, C. G. Antonopoulos, and C. Hens, "Emergence of mixed mode oscillations in random networks of diverse excitable neurons: The role of neighbors and electrical coupling," *Frontiers in Computational Neuroscience*, vol. 14, p. 49, 2020, ISSN: 1662-5188. DOI: 10.3389/fncom.2020.00049. [Online]. Available: https://www.frontiersin.org/article/10.3389/fncom.2020.00049.

[40] J. Marro and J. Torres, "Chaotic hopping between attractors in neural networks," *Neural Networks*, vol. 20, pp. 230–235, Apr. 2006.

[41] J. P. Roach, A. Pidde, E. Katz, J. Wu, N. Ognjanovski, S. J. Aton, and M. R. Zochowski, "Resonance with subthreshold oscillatory drive organizes activity and optimizes learning in neural networks," *Proceedings of the National Academy of Sciences*, vol. 115, no. 13, E3017–E3025, 2018, ISSN: 0027-8424. DOI: 10.1073/pnas.1716933115. eprint: https://www.pnas.org/content/115/13/E3017.full.pdf. [Online]. Available: https://www.pnas.org/content/115/13/E3017.

[42] C. Cakan and K. Obermayer, "Biophysically grounded mean-field models of neural populations under electrical stimulation," *PLOS Computational Biology*, vol. 16, no. 4, pp. 1–30, Apr. 2020. DOI: 10.1371/journal.pcbi.1007822. [Online]. Available: https://doi.org/10.1371/journal.pcbi.1007822.

[43] M. A. Buice, J. D. Cowan, and C. C. Chow, "Systematic Fluctuation Expansion for Neural Network Activity Equations," *Neural Computation*, vol. 22, no. 2, pp. 377–426, Feb. 2010, ISSN: 0899-7667. DOI: 10.1162/neco.2009.02-09-960. eprint: https://direct.mit.edu/neco/article-pdf/22/2/377/831286/neco.2009.02-09-960.pdf. [Online]. Available: https://doi.org/10.1162/neco.2009.02-09-960.

[44] L. Hertäg, D. Durstewitz, and N. Brunel, "Analytical approximations of the firing rate of an adaptive exponential integrate-and-fire neuron in the presence of synaptic noise," *Frontiers in Computational Neuroscience*, vol. 8, p. 116, 2014, ISSN: 1662-5188. DOI: 10.3389/fncom.2014.00116. [Online]. Available: https://www.frontiersin.org/article/10.3389/fncom.2014.00116.

[45] M. Augustin, J. Ladenbauer, F. Baumann, and K. Obermayer, "Low-dimensional spike rate models derived from networks of adaptive integrate-and-fire neurons: Comparison and implementation," *PLOS Computational Biology*, vol. 13, no. 6, pp. 1–46, Jun. 2017. DOI: 10.1371/journal.pcbi.1005545. [Online]. Available: https://doi.org/10.1371/journal.pcbi.1005545.

[46] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The spinnaker project," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652–665, 2014. DOI: 10.1109/JPROC.2014.2304638.

[47] M. Bouvier, A. Valentian, T. Mesquida, F. Rummens, M. Reyboz, E. Vianello, and E. Beigne, "Spiking neural networks hardware implementations and challenges," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 15, no. 2, pp. 1–35, Jun. 2019, ISSN: 1550-4840. DOI: 10.1145/3304103. [Online]. Available: http://dx.doi.org/10.1145/3304103.

[48] A. Salles and M. Farisco, "Of ethical frameworks and neuroethics in big neuroscience projects: A view from the hbp," *AJOB Neuroscience*, vol. 11, no. 3, pp. 167–175, 2020, PMID: 32716744. DOI: 10.1080/21507740.2020.1778116. eprint: https://doi.org/10.1080/21507740.2020.1778116. [Online]. Available: https://doi.org/10.1080/21507740.2020.1778116.

[49] E. M. Izhikevich and G. M. Edelman, "Large-scale model of mammalian thalamocortical systems," *Proceedings of the National Academy of Sciences*, vol. 105, no. 9, pp. 3593–3598, 2008, ISSN: 0027-8424. DOI: 10.1073/pnas.0712231105. eprint: https://www.pnas.org/content/105/9/3593.full.pdf. [Online]. Available: https://www.pnas.org/content/105/9/3593.

[50] N. Block, "On a confusion about a function of consciousness," *Behavioral and Brain Sciences*, vol. 18, no. 2, pp. 227–247, 1995. DOI: 10.1017/S0140525X00038188.

[51] A. M. Turing, "Computing machinery and intelligence," *Mind*, vol. 59, no. October, pp. 433–60, 1950. DOI: 10.1093/mind/LIX.236.433.

[52] W. Singer, "Consciousness and the structure of neuronal representations," *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, vol. 353, no. 1377, pp. 1829–1840, 1998. DOI: 10.1098/rstb.1998.0335. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rstb.1998.0335. [Online]. Available: https://royalsocietypublishing.org/doi/abs/10.1098/rstb.1998.0335.

[53] N. Emmerich, "This is the age of the brain – but bending beliefs and feelings raises political questions," *The Conversation*, May 2015.

[54] A. Destexhe and D. Contreras, "The fine structure of slow-wave sleep oscillations: From single neurons to large networks," in. Jan. 2011, pp. 69–105, ISBN: 978-1-4614-0172-8. DOI: 10.1007/978-1-4614-0173-5_4.

[55] J. Copeland and P. Dillon, "The health and psycho-social consequences of ketamine use," *International Journal of Drug Policy*, vol. 16, no. 2, pp. 122–131, 2005, Special Focus:The Taliban and Opium, ISSN: 0955-3959. DOI: https://doi.org/10.1016/j.drugpo.2004.12.003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0955395905000101.

[56] A. Khakhalin and A. Frolov, "[joint application of the independent component analysis and the nonstationary fluctuation analysis for the investigation of early stages of long-term potentiation in the rat hippocampus]," *Zhurnal vysshei nervnoĭ deiatelnosti imeni I P Pavlova*, vol. 55, pp. 293–304, May 2005.

[57] R. Brette, *What is wrong with computational neuroscience?* http://romainbrette.fr/what-is-computational-neuroscience-xvii-what-is-wrong-with-computational-neuroscience/, 2013.

[58] R. Naud, N. Marcille, C. Clopath, and W. Gerstner, "Firing patterns in the adaptive exponential integrate-and-fire model," *Biological Cybernetics*, vol. 99, pp. 335–347, 2008.