Bachelor Thesis

April 12, 2020

A Multidisciplinary Analysis of the Link between Structure and Function in Biological Neural Networks

Skander Moalla

Internship Tutor:

Prof. Kathryn Hess

Referent Instructor:

Prof. Pierre-Yves Strub

It is through science that we prove,
but through intuition that we discover.
— Henri Poincaré

To the beauty of science and the excitement of multidisciplinary projects.

Acknowledgements

First, I would like to thank my supervisor Prof. Kathryn Hess. Thank you for welcoming me to your research group and for giving me the opportunity to pursue such an exciting multidisciplinary research experience at the Blue Brain Project. Thank you for integrating me into the lab's agenda by inviting me to the regular brainstorming sessions. Thank you for feeling always available despite your hectic schedule. And thank you for your constant encouragement and support. I would also like to thank everyone in the lab, especially Nicolas Ninin who welcomed me on my first day at the Blue Brain, Daniela Egas Santander who constantly reminded me of how complex neural networks are, Stefania Ebli, Adélie Garin, and Celia Hacker. I have certainly learned from each one of you. I would like to thank Abel Sagodi, my fellow intern, for always being there to hear my early conjectures and start endless philosophical digressions. I would like to thank the HR team at the Blue Brain Project for providing excellent working conditions. I would like to thank Coralie Link and the rest of the team at the Student Exchange Office of EPFL for their help with the exchange procedures. I would like to thank the administration of the Bachelor program of Ecole Polytechnique for its flexibility and for allowing us to do our theses abroad. In particular, I would like to thank Céline Chalard and Sophie Ginisty for helping me secure the internship and getting the scholarship of the Swiss-European Mobility Programme. I would like to thank Prof. Emmanuel Haucourt without who this internship would not have existed. Thank you for putting me in contact with Prof. Kathryn Hess. Finally, I would like to thank my family for their constant support, especially during this lockdown period.

S. M.

Abstract

In recent applications of algebraic topology to explore and analyze digital reconstructions of neocortical microcircuitry developed at the Blue Brain Project, researchers have been able to quantify the structural complexity of a neural network and its activity. Those results suggest a correlation between the structural complexity and the functional complexity of a neural network. In this report, we first provide an overview of the multidisciplinary approach one has to adopt to study such a correlation. Incidentally, we build the simple framework of choice: an implementation of a simple yet very efficient neuron model using Brian — a user-friendly python simulator for spiking neurons. Then, we focus on the topological structures suspected to drive this correlation, namely, directed cliques of neurons and build a network where this correlation can be measured and analyzed using tools from information theory. In particular, we look at the evolution of mutual information between pairs of neurons within cliques and break this measure down using the partial information decomposition. The results of our analysis suggest that when subject to a stimulus, high dimensional cliques transform the information coming from the stimulus to generate new information reflecting their high degree of organization.

Contents

1 Introduction

Understanding how the structure of a neural network, formed by the connections between its neurons, affects its emergent behavior and function has been a major challenge in neuroscience. Multiple disciplines have been applied alongside neuroscience to address this question. In graph theory, for example, researchers have defined several measures describing a network's *segregation*, the degree to which its elements form separate clusters; *integration*, the capacity of a network as a whole to become interconnected; or *influence*, the individual contribution of some nodes to the network's structural integrity and information flow (Sporns, 2013). Other measures such as *efficiency* introduced by Latora and Marchiori (2001) attempt to describe at the same time both the local and global efficiency of a network to exchange information. In the context of efficiency, biological neural networks are seen as small-world networks. Nevertheless, these measures fail to describe most local biological network properties, such as the different roles of individual neurons or relatively small groups of neurons. Other attempts, combining dynamical systems and graph theory, try to predict neural network dynamics via graphical analysis (Morrison and Curto, 2019; Curto et al., 2019, 2018). Although promising, these results are still constrained to fairly theoretical neural network models that are far from resembling biological neurons.

More recently algebraic topology (Hatcher, 2002) has been applied to digital reconstructions of rat neocortical microcircuitry developed by the Blue Brain Project. Reimann et al. (2017) developed a mathematical framework to analyze both the structural and functional topology of neural networks that integrates local and global descriptors, thus allowing the authors to identify a relationship between them. Applying this framework to the digital reconstructions, the authors found a remarkably high number and variety of specific high dimensional topological structures, namely directed cliques and cavities. *Directed cliques* are groups of neurons that are all-to-all connected and have no cycle, thus having a clear direction for the flow of information through the neurons, and *cavities* arise when directed cliques bind appropriately by sharing neurons, but without forming a larger clique due to missing connections. We define these structures more formally in section 1.2. This high number of directed cliques and cavities had never been seen before in biological or artificial neural networks and is far greater

than the numbers found in various null models of directed networks. Furthermore, when simulating the microcircuit activity in response to sensory stimulus, the authors observed that pairwise correlations in neural activity increased with the number and dimension of the directed clique to which a pair of neurons belongs suggesting that correlated activity binds neurons into high-dimensional active cliques. The main goal of our research will be to further understand how these topological structures influence emergent dynamics.

In this report, we first provide an introduction to the three disciplines we had to learn to tackle the challenge of understanding how the brain works and build on the results of Reimann et al. (2017), namely neuroscience, algebraic topology, and information theory.

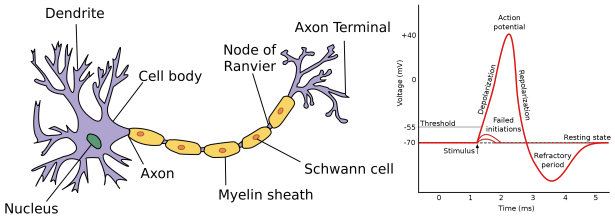
Secondly, we present the *simple framework of choice*, our initiative to combine Izhikevich's simple neuron — a simple yet very efficient neuron model — with a user-friendly python simulator for spiking neurons called Brian. We believe that this framework could be a very useful tool used by scientists to run elementary simulations and conduct new kinds of analyses of neural networks. In our case, we use our framework to apply a novel analysis of neural networks mixing information theory and algebraic topology. Bringing together the benefits of both Izhikevich's simple neuron and Brian, our framework is easy to use, computationally efficient and accurate, and easy to review. It allows scientists to spend most of their time on tuning the details of their model and performing their novel analysis, rather than on the intricacies of the implementation of neural network simulations.

Lastly, we examine the local topological structures studied by Reimann et al. (2017), namely directed cliques of neurons, that are suspected to drive the correlation between the structure and the function of a neural network. We build networks of isolated cliques of neurons of increasing dimension which allows us to study more directly the effect of topology on emergent behavior. As a novel approach to (Reimann et al., 2017), we use tools from information theory such as mutual information and the partial information decomposition rather than the widely used Pearson correlation to study the interactions between neurons in a clique. We find that when subject to a stimulus, high dimensional cliques transform the information coming from the stimulus to generate new information reflecting their high degree of organization. Precisely, the information shared by the last neurons in the cliques is initially lower than the information encoded about the stimulus, however, the former information eventually outweighs the latter information coming from the stimulus as the dimension of the clique grows. Furthermore, decomposing this information into four disjoint partial components, we find that the information generated by the last neurons of the clique is created synergistically with the information coming from the stimulus, whence our hypothesis that cliques of neurons transform information coming from a stimulus.

1.1 Neuroscience

The first step in analyzing the relationship between the structure and the function of the brain is to understand how individual neurons work and how they connect and exhibit complex behaviors.

1.1.1 Basics: Neurons, Spikes, and Synapses



(a) A neuron (b) A spike

Figure 1.1 – Figures from wikipedia.org

From a computational point of view, a typical neuron consists of three main parts: the soma, dendrites, and the axon. The *soma* is the cell body, *dendrites* are the input ends of the neuron, and the *axon* is the output end (Figure 1.1a).

A neuron connects to its neighbors via *synapses*. At the farthest tip of its axon's branches are axon terminals where the neuron, referred to as the *presynaptic* neuron, can transmit a signal across the synapse to another cell, referred to as the *postsynaptic* neuron.

Inputs received by a neuron through the contacts of its dendritic tree produce electrical currents that change its membrane potential (Figure 1.1b). These changes, resulting from synaptic currents, are called *postsynaptic potentials* (PSPs). Small currents produce small PSPs and larger currents induced from a combination of several other neurons are amplified by the neuron which generates an *action potential* or a *spike*. For regular spiking neurons, this spiking happens when the input reaches a certain threshold (Figure 1.1b).

The behavior of a single neuron may be easy to predict, but when multiple neurons of various types connected via various synapses span a large neural network, the emergent behavior of such network becomes complex and very hard to predict.

A natural question that we had to address early in our research was how could we simulate a neural network? Which model could we use? Which simplifications could we make? And which simplifications would lessen the significance of our analysis?

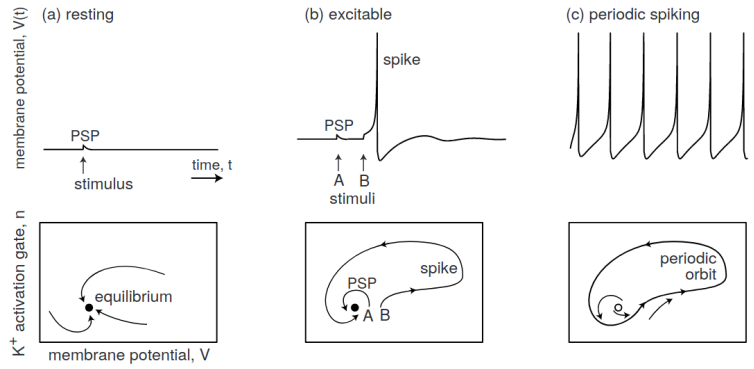
There is a wide spectrum of neuron models studied in the literature. The most complex ones include the Hodgkin–Huxley model (Hodgkin and Huxley, 1952) which describes neuronal dynamics in terms of activation and inactivation of voltage-gated conductances and the

simplest ones include the leaky integrate-and-fire model which can be described with a single differential equation. Fortunately, we will see in the next section that harmony between computational simplicity and biological accuracy is achievable.

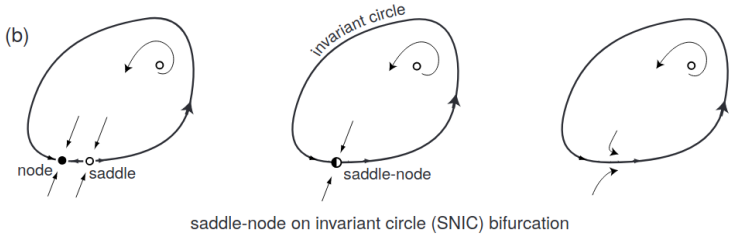
1.1.2 Simple Model of Choice

In (Izhikevich, 2007), the author provides a dynamical systems perspective on neurons. His main point is that a neuron should be viewed as a nonlinear dynamical system in addition to its more common representations in terms of ions and channels by biologists or in terms of input/output relationships by theoreticians.

The advantage of such a perspective is that a model need not be complex (e.g. with a variable for each electrophysiological parameter) to be considered accurate. Indeed, in the dynamical systems approach to neuroscience, the focus is on *phase portraits* and *bifurcations* rather than on equations.



(a) Phase portraits of a neuron in resting, excitable, and periodic spiking activity.



(b) Bifurcation of an equilibrium state leading to the transition from resting to periodic spiking activity of a neuron.

Figure 1.2 – Phase portrait and bifurcation. Figures 1.9 and 1,12a from (Izhikevich, 2007).

A *phase portrait* is a sketch used to qualitatively analyze a dynamical system. It depicts all stable and unstable equilibria (as black and white circles, respectively), representative trajectories, and corresponding attraction domains in a system's state. In Figure 1.2a for example, the resting state (a) of the neuron corresponds to a stable equilibrium (black circle)

since inward and outward currents are balanced and small perturbations or membrane noise do not affect the state of the neuron. In its excitable state (b) large perturbations such as the stimulus B are amplified by the neuron and result in a spike. Finally, the injection of a very strong current in the neuron puts it in its periodic spiking state (c) where the resting equilibrium becomes unstable (white circle).

Bifurcations denote a qualitative change in the phase portrait of a system. In our example (Figure 1.2a), going from the resting to the excitable state does not correspond to a bifurcation since the neuron ends its trajectory at its stable equilibrium. However, there is a bifurcation going from the excitable to the periodic spiking state, as the equilibrium becomes unstable and a periodic orbit appears. In addition, Figure 1.2b shows the steps of another type of bifurcation exhibited by a regular-spiking neuron as we stimulate it with an increasing current.

In this context, a model is valid if it exhibits the same bifurcations as the neuron under consideration. This way, Izhikevich builds what he calls the *simple model of choice* for a neuron by starting with a one-dimensional linear ODE and gradually increasing the complexity of this model until it becomes the simplest model able to reproduce the neurocomputational properties of 20 of the most fundamental biological neurons.

At its core, the model:

$$\dot{v} = v^2 - u + I \qquad \text{if } v \geq 1, \text{ then} \tag{1.1}$$

$$\dot{u} = a(bv - u) \qquad v \leftarrow c, \quad u \leftarrow u + d \tag{1.2}$$

is a two-dimensional system consisting of a fast variable v and a slower recovery variable u. In addition, the model has four dimensionless parameters $a, b, c,$ and d which allow the model to reproduce different dynamics of biological neurons. The variable I is the input received by the neuron.

From a biological perspective, it is more convenient to use the model in the form:

$$C\dot{v} = k(v - v_r)(v - v_t) - u + I \qquad \text{if } v \geq v_{peak}, \text{ then} \tag{1.3}$$

$$\dot{u} = a[b(v - v_r) - u] \qquad v \leftarrow c, \quad u \leftarrow u + d \tag{1.4}$$

where the variables can be interpreted as:

Variables	Dimensionless parameter	Biological / tuning parameters
v: membrane potential	a: recovery time constant	C: membrane capacitance
u: recovery current	b: constant	v_r: resting membrane potential
	c: potential reset value	v_t: instantataneous threshold potential
	d: constant	v_{peak}: spike cutoff
		k: constant

This form is equivalent to 1.1. The parameters $C, v_r, v_t, v_{peak},$ and k are used to fine-tune the model.

Although a simpler model such as the leaky integrate-and-fire:

$$\dot{v} = I - v \qquad \text{if } v \geq v_{peak} \text{ then } v \leftarrow v_{reset} \tag{1.5}$$

could be used by mathematicians seeking to prove theorems, Izhikevich considers that this model is a waste of time for computational neuroscientists who wish to simulate large-scale networks. Indeed, the integrate-and-fire model does not reproduce the correct bifurcation in its transition from resting to repetitive spiking. which results in a logarithmic scaling of its F-I curve (frequency response (F) to different injected current (I)) as opposed to the expected square-root scaling (Figure 2.1c). In addition, the simple model of choice is less prone to numerical errors affecting its spike timing since these errors are inversely proportional to the slope of the membrane voltage at the reset step (Figure 1.3).
Finally, Izhikevich does not consider the integrate-and-fire model as a *true* spiking model, since the event of a *spike* in this model only denotes that the membrane reached the threshold and was reset. The model cannot generate a fast upstroke (Figure 1.3).

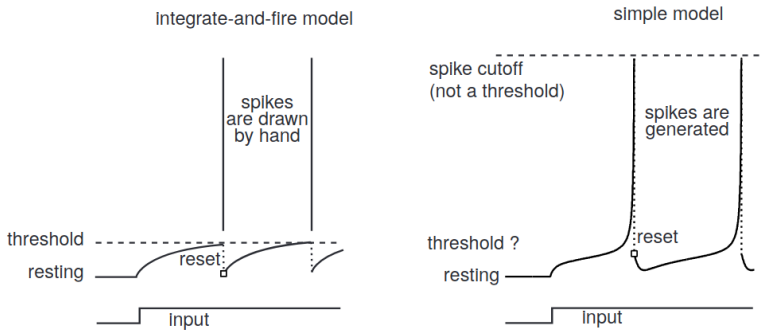


Figure 1.3 – The integrate-and-fire model cannot generate a spike and the slope of its membrane potential near the reset is small. On the contrary the slope near the rest of the simple model is almost infinite. Figure 8.7 from (Izhikevich, 2007)

This motivates us to use the simple model of choice in our simulations while investigating the link between the topological complexity and the functional complexity of neural networks. Moreover, in our attempt to find a convenient simulation software we incidentally build the simple framework of choice in section 2.1 which is an implementation of the simple model of choice using Brian (Goodman and Brette, 2009), a user-friendly python simulator for spiking neurons.

1.1.3 Brian Simulator

Brian (Goodman and Brette, 2009) is a python simulator for spiking neural networks. It is built with the goal of making the writing of simulation code easy, quick, and flexible. This way, scientists can focus on the details of their models rather than on their implementation.

In the following, we introduce the basic features and objects of Brian that we use to implement the simple framework of choice in 2.1.

Defining a model

With Brian, one defines any desired model directly by providing its mathematical definition, consisting of differential equations and discrete events (the effect of spikes) written in standard mathematical notation. This is done via the object `NeuronGroup` (Figure 1.4).

Synapses between neurons can then be added in similar way with the object `Synapses`, to which one provides the `NeuronGroup`s to be connected and the model of the synapses. The details of the connections can then be specified via the method `Synapses.connect()`.

```
tau = 1*ms                              # A constant time variable
VT = 0.8                                # The threshold value
VR = 0                                  # The reset value
I = 1                                   # A constant input
nb_neurons = 10                          # The number of neurons

model = 'dV/dt = (I - V)/tau : 1'       # The equation defining the neuron model
threshold = 'V > VT'                    # The discrete event triggering a spike
reset = 'V = VR'                        # The discrete event to execute after a spike

neurons = NeuronGroup(nb_neurons, model, threshold=threshold, reset=reset)
connections = Synapses(neurons, neurons, on_pre='V_post += 0.2')
connections.connect(condition='i < j') # The details of the connections

spikemon = SpikeMonitor(neurons)        # Record the spike time of each neuron
statemon = StateMonitor(neurons, 'V')  # Record the evolution of the variable V for each neuron

run(100*ms)                             # Run the simulation
```

Figure 1.4 – Illustration of the basic objects of Brian with the definition of a network of 10 leaky integrate-and-fire neurons 'dV/dt = (I - V)/tau'. The connection condition 'i < j' allows to connect the neurons to form a directed clique.

Running and recording a simulation

Once neurons and synapses are defined, a `SpikeMonitor` and `StateMonitor` can be added to the network to record respectively the spike timing of each neuron and the time evolution of any variable defined in the neuron model (Figure 1.4).

When every variable is defined a simple call to the function `run(duration)`, with the `duration` of the simulation as an argument, is enough to run and record the simulation.

In our information-theoretic analysis we leverage the decorator `@network_operation` to

regularly run a user-defined python function that allows us to flush the data recorded in a time bin and initialize the containers of the following one.

The reader can refer to Stimberg et al. (2019) for a more comprehensive presentation and review of the capabilities of Brian.

1.1.4 Next Level: The Blue Brain Project

The Blue Brain Project was founded in 2005. Its goal is to build biologically detailed digital reconstructions and simulations of the rodent brain and, ultimately, the human brain.

In (Markram et al., 2015), the authors present a digital reconstruction of the microcircuitry of rat somatosensory cortex built at the Blue Brain Project. In this reconstruction neurons are positioned in a three-dimensional virtual volume respecting experimentally observed data, they are objectively classified into 55 morphological cell types, and they are modeled using the Hodgkin-Huxley equation. Moreover, connections between the neurons happen where their dendrites touch each other in the virtual space and when other constraints are satisfied. The synaptic transmission is also modeled integrating extensive experimental data. That is to say, this reconstruction closely resembles the biological tissue in terms of the number, types, and densities of neurons and their synaptic connectivity. Furthermore, simulations of this reconstruction reproduce multiple emergent experimental behaviors without specific parameter tuning.

This reconstruction allows for novel analysis in the field of neuroscience to happen such as (Reimann et al., 2017) and enables the formulation of detailed predictions of behaviors experimentally not observed yet. To date, The Blue Brain Project has achieved a reconstruction of the mouse whole-neocortex micro-connectome (Reimann et al., 2019). We believe this is an incredibly important moment in the history of neuroscience. After years of theory and experimentation, neuroscience is joining many other sciences and engineering fields in a phase where predictions are made via simulations and, eventually, where interpretations of experiments are validated via extensive simulations.

1.2 Algebraic Topology

For the last decade, and especially for the past five years, the number of applications of algebraic topology in neuroscience has been greatly increasing. For example, algebraic topology has been used to predict the dynamical regime of a neural network (Bardin et al., 2019) and has played a key role in building an objective morphological classification of neocortical pyramidal cells (Kanari et al., 2018, 2019) for which a consensus on the number had not been reached before.

Such applications promote a continuous improvement of the reconstruction of the neocortical microcircuitry built by the Blue Brain Project (Markram et al., 2015) as well as better identification of the relationship between the structural and the functional complexity of neural networks. Indeed, in (Reimann et al., 2017), the authors developed a mathematical framework to analyze the structural and the functional topology of neural networks and quantify their complexity. This analysis suggests a correlation between structure and function that we attempt to identify in section 2.2.

In this section we introduce the fundamental concepts of algebraic topology used in computational neuroscience such as homology. We also provide an overview of the evolution of the software tools developed to compute the topological invariants of a network and the challenges faced in this field. Finally, we mention some applications of topological data analysis (TDA) in machine learning that may be of interest to our fellow classmates doing machine learning.

1.2.1 Simplicial Complexes

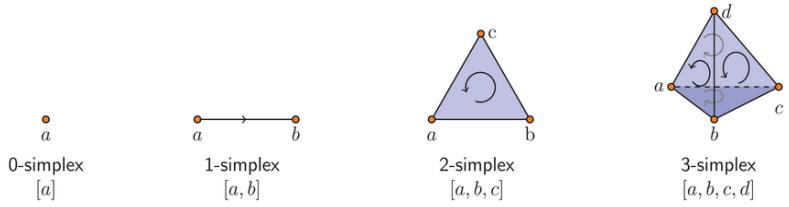


Figure 1.5 – k-simplices with given orderings. Figure 1 from (Topaz et al., 2015).

One can think of a *simplicial complex* as a higher dimensional analog of a graph, with its building blocks referred to as *simplices*. A 0-simplex and a 1-simplex are the analogs of respectively a vertex and an edge and a higher dimensional building block formed by $n+1$ vertices is called an n-simplex, which is geometrically the convex hull of its $n+1$ affinely positioned vertices. e.g. a 2-simplex is a filled triangle, a 3-simplex is a filled tetrahedron, and so on. An n-simplex formed by the vertices (or 0-simplices) $v_0, \ldots, v_n$ can be denoted by $\{v_0, \ldots, v_n\}$, however, it is convenient [1] to set an ordering of the vertices of such n-simplex, which will thus be denoted by the ordered set $(v_0, \ldots, v_n)$. A (proper) *face* of a simplex $\sigma :=$

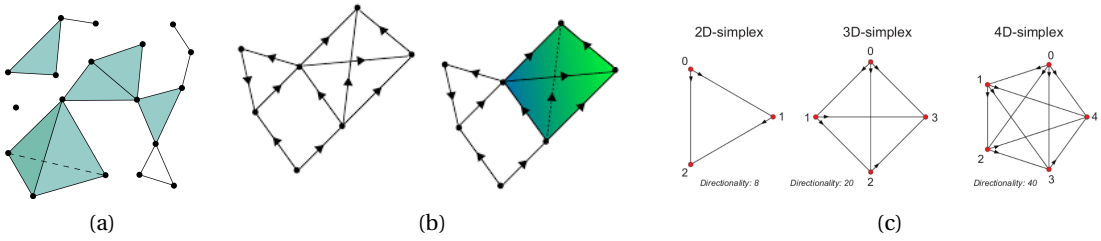
[1] for the sake of defining homology and associating a directed graph to a simplicial complex.

$(v_0,\ldots,v_n)$ is a simplex τ with vertices any nonempty (proper) subset of $(v_0,\ldots,v_n)$; it inherits its ordering naturally from the ordering of σ. The i-th face of $\sigma := (v_0,\ldots,v_n)$ is the simplex σ^i obtained by removing the vertex v_{n-i}.

A simplicial complex is then defined as a collection $\mathscr{S}$ of simplices with the property that if a simplex $\sigma \in \mathscr{S}$, then every face of σ is also in $\mathscr{S}$. We write the set of all k-simplices of $\mathscr{S}$ as $\mathscr{S}_k$.

Directed graphs give rise to simplicial complexes in a natural way. Precisely, if we consider a directed graph $G = (V,E)$, where V is the set of vertices and E is the set of directed edges consisting of ordered pairs $(v_1,v_2) \in V \times V$, then G can be associated to the simplicial complex $\mathscr{S}$, where $\mathscr{S}_0 = V$ and whose n-simplices $\mathscr{S}_n$ for $n \geq 1$ are the $(n+1)$-tuples $(v_0,\ldots,v_n)$ of vertices such that for each $0 \leq i < j \leq n$, $(v_i,v_j) \in E$. In this context, the vertices v_0 and v_n are called respectively the *source* and the *sink* of the simplex $(v_0,\ldots,v_n)$, as there is an edge directed from v_0 to v_i for all $0 < i \leq n$ edge directed from v_i to v_n for all $0 \leq i < n$.

Borrowing the terminology of *clique* from graph theory denoting a set of vertices that are all-to-all connected, we use the term *directed cliques* to refer to ordered sets of vertices $(v_0,\ldots,v_n)$ that are all-to-all connected such that there is an edge from a vertex v_i to another vertex v_j if an only if $i < j$. It is then straightforward to see that the directed cliques in a graph can be associated to the simplices in the simplicial complex induced by this same graph. In this regard, we will use the terms *directed clique* and *simplex* interchangeably in this work.



(a) (b) (c)

Figure 1.6 – (a) Simplicial complex. Figure from wikipedia.org. (b) Simplicial complex induced by a directed graph. Figure by Paweł Dłotko. (c) Directed cliques. Figure S2 from (Reimann et al., 2017).

1.2.2 Simplicial Homology

Homology serves to measure the "topological complexity" of simplicial complexes via numerical quantities it induces such as *Betti numbers*. *Homology groups* are algebraic objects one can associate with simplicial complexes, in which context one speaks about *simplicial homology*. In what follows, we provide an elementary description of homology and its basic properties as it is outlined by (Reimann et al., 2017) in section 4.1 The topological toolbox. In a similar way, we restrict our study to *mod 2 simplicial homology*.

Denoting $\mathbb{F}_2$ the field of two elements and $\mathscr{S}$ a simplicial complex, we define $C_n(\mathscr{S},\mathbb{F}_2)$ as

the $\mathbb{F}_2$-vector spaces whose basis elements are the $n-$simplices $\sigma \in \mathscr{S}_n$ for each $n \geq 0$. These spaces form the *chain complex* $C_*(\mathscr{S}, \mathbb{F}_2) := \{C_n = C_n(\mathscr{S}, \mathbb{F}_2)\}_{n \geq 0}$.

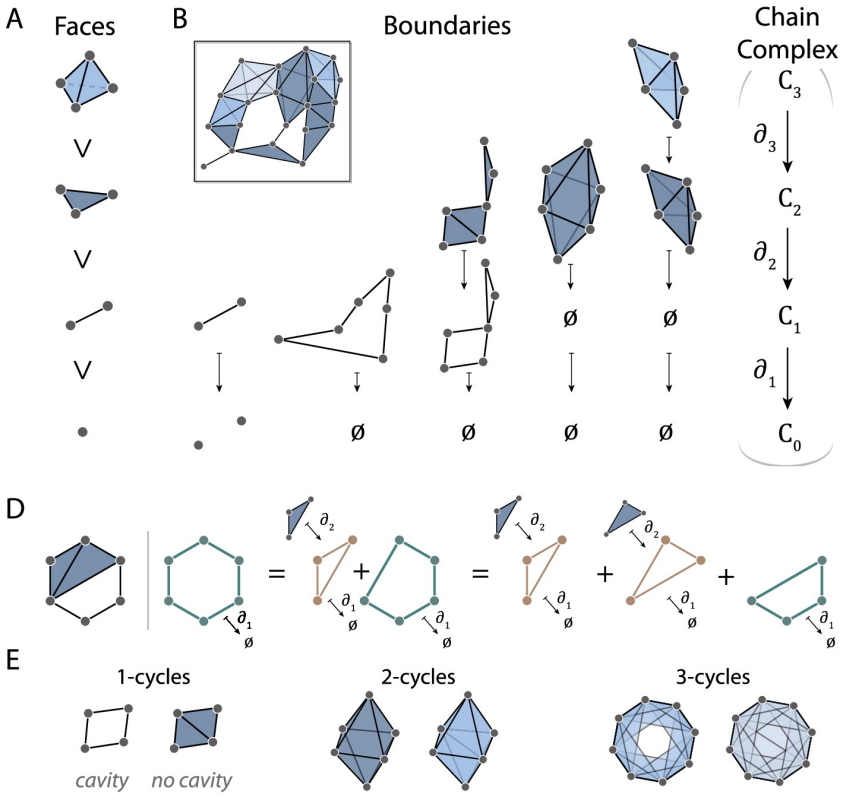


Figure 1.7 – Chain complexes and homology. (A) A 3-simplex (top) has 2-simplices, 1-simplices, and 0-simplices as faces. (B) Boundary maps take k-chains to their boundaries. Examples shown for dimensions 1 through 3. These boundary maps connect the chain groups forming the chain complex (right). (D) Given the simplicial complex shown at the (left), the three green cycles are equivalent because they differ by boundary cycles (gold). (E) Examples of k-cycles for k = 1 (left), k = 2 (middle), and k = 3 (right). For each k we show a non-trivial (cavity-enclosing) cycle on the left and a trivial cycle on the right. Figure 4 from (Sizemore et al., 2018)

For each $n \geq 1$, we then define the *boundary map* $\delta_n : C_n \mapsto C_{n-1}$ which maps an n-simplex σ to the formal sum of its faces $\delta_n(\sigma) = \sigma^0 + \sigma^1 + \cdots + \sigma^n$. The definition of the boundary map on the basis elements of C_n then naturally extends to all the elements in C_n. This allows us to define the *homology group* $H_n(\mathscr{S}, \mathbb{F}_2) = \text{Ker}(\delta_n)/\text{Im}(\delta_{n+1})$ for $n \geq 1$ and $H_0(\mathscr{S}, \mathbb{F}_2) = C_0/\text{Im}(\delta_1)$. Intuitively it refers to the group formed by elements that are a combination of simplices in a given dimension forming a "cycle" that is not filled by simplices in a larger dimention, whence the term *cavity*. The n-th *Betti number* $\beta_n(\mathscr{S})$ of a simplicial complex measures the number of such cavities formed by simplices of dimension n. It is defined as the the dimension of the n-th homology group $H_n(\mathscr{S}, \mathbb{F}_2)$.

1.2.3 Computing Homology

Computing the homology of a simplicial complex can be translated to a problem of numerical linear algebra. Denoting $|\mathscr{S}_n|$ the the number of n-simplices in a simplicial complex $\mathscr{S}$, the boundary map δ_n can be modeled as a $(|\mathscr{S}_{n-1}| \times |\mathscr{S}_n|)$-matrix D_n with entries in $\mathbb{F}_2$. The $n-$th Betti number is then simply given by the equation $\beta_n(\mathscr{S}) = \text{null}(D_n) - \text{rk}(D_{n+1})$.

Despite this straightforward conceptual result, several computational challenges are still to be addressed. In 2017, the computation of the homology of the simplicial complex induced by the microcircuit formed by neurons 31,000 neurons was completely out of reach for the supercomputer of the Blue Brain Project. Shortly after, in (Luetgehetmann et al., 2019), the authors presented a new computing package, *Flagser*, that allowed the computation of the homology of this reconstruction. This package even enabled the approximate computation of the homology of the Blue Brain Project's recent reconstruction of a full region of a mouse's neocortex having twice as many neurons as the previous reconstruction and over 1.2 trillion 11-dimensional simplices.

1.2.4 Topological Data Analysis in Machine Learning

Anyway, machine learning is just an extension of neuroscience.
— Abel Sagodi

In some cases, it is important to understand the underlying characteristic shapes on the images of a data set or to have specific feature sets encoding them in order to perform accurate classifications. In this context, TDA allows us to extract topological features from images that can be combined with machine learning techniques to form a powerful pipeline leveraging these characteristic shapes.

For example, Garin and Tauzin (2019) show that TDA techniques provide powerful dimensionality reduction algorithms for images by performing a classification of the MNIST data set with 5 times fewer features than the grayscale pixel value features. These features come from a technique called *persistent homology* which, as homology, extracts the numbers of connected components, cycles, and cavities (captured by the Betti numbers) as well as their birth and death during an iterative process called a *filtration*.

We previously discussed how homology could be applied to directed graphs. In a similar way, persistent homology can be applied to *weighted* directed graphs. In addition, persistent homology can be applied to point clouds via their Vietoris–Rips or Čech construction and to images via their cubical complex construction.

For a complete introduction to the computation of persistent homology coupled with real-world applications we suggest (Otter et al., 2017) and for an overview of the evolution of persistent homology, starting with the first intuition developed 20 years ago to the more modern categorical point of view, we suggest (Perea, 2018)

1.3 Information Theory

Very often, the data studied in neuroscience experiments is multivariate and the interactions between the variables are nonlinear, not to mention that the set of possible interactions between variables is very large. Because it can address these challenges, information theory is a very useful analysis tool in neuroscience.

Indeed, information theory is model-independent, i.e. no hypothesis regarding the structure of the interactions between the variables is required, it can be applied to any mixture of data types e.g. data consisting of the state of a square pulse stimulus and the number of spikes of a neuron in some time interval, and it can detect linear and nonlinear interactions.

Furthermore, Li and Li (2019) provide a proof of the suitability of such measures of information for neural networks. Using a simplified model of the neuron, with discrete membrane potential, they were able to prove stochastic stability and the law of large numbers on these measures.

In this section, we introduce the information-theoretic measures we use in section 2.2 to analyze the effects of topological structure on the interactions between neurons.

1.3.1 Information, Uncertainty, and Entropy

One way [2] to intuitively define information is to say that one variable provides information about another variable when knowledge of the first, in expectation, reduces uncertainty in the second [3](Cover and Thomas, 2006). In this perspective, we define *entropy* as a measure of the uncertainty contained in a variable.

Precisely, given a discrete random variable X with outcomes $x_1, \ldots, x_n$, each with probability $p(x_i)$, we define the entropy $H(X)$ of X as: (Shannon, 1948; Cover and Thomas, 2006)

$$H(X) = \sum_{x \in \Omega} p(x) log_2 \left(\frac{1}{p(x)} \right), \quad \text{where} \quad \Omega = \{x_1, \ldots, x_n\}$$

Similarly, for two discrete random variables X and Y with outcomes in respectively Ω_X and Ω_Y and a joint probability distribution p we can define the joint entropy $H(X,Y)$ as: (Cover and Thomas, 2006)

$$H(X,Y) = \sum_{x \in \Omega_X, y \in \Omega_Y} p(x,y) log_2 \left(\frac{1}{p(x,y)} \right)$$

Finally, we can define the conditional entropy, which defines the expected uncertainty in a variable given the state of another variable. The entropy $H(X|Y)$ of X conditioned on Y is

[2] Another way would be to define the self-information of an outcome of a random variable. In this perspective, entropy would be the expected (self-) information in a variable.

[3] Precisely, this is what we will define later as *mutual information*.

given by: (Cover and Thomas, 2006)

$$H(X|Y) = \sum_{x \in \Omega_C, y \in \Omega_Y} p(x,y) log_2 \left(\frac{1}{p(x|y)} \right)$$

1.3.2 Mutual Information

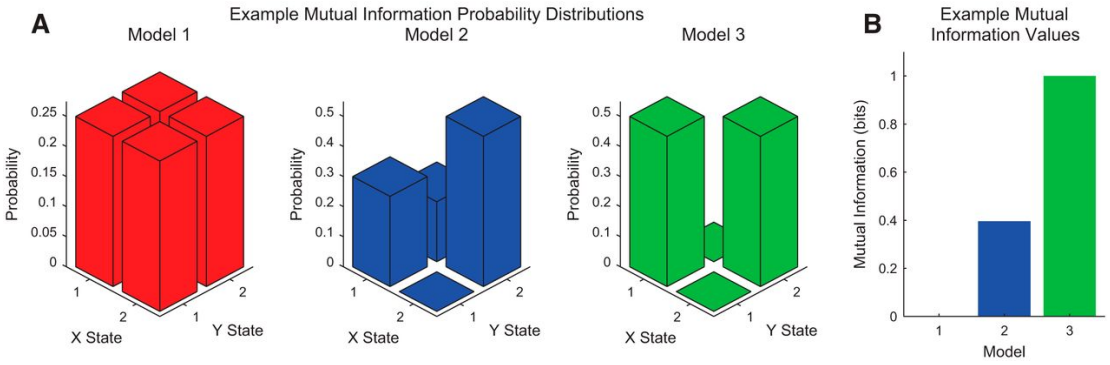


Figure 1.8 – Example of mutual information computations. A. Probability distributions for three models (red, blue, and green). B. their associated mutual information values. In model 1, the X and Y variables are independent, so their mutual information is zero. In model 2, knowledge of X or Y reduces uncertainty in the state of the other variable to some extent, so nonzero mutual information is observed. In model 3, X and Y are identical, so their mutual information is maximal. Figure 4 from (Timme and Lapish, 2018)

Recalling that information is a reduction of uncertainty, we can say that if learning the state of one variable Y reduces the uncertainty in another variable X in expectation, then the first variable provides information about the second variable. Since we quantified uncertainty using entropy, we can also quantify this reduction in uncertainty also in terms of entropy, which gives a measure for information. We refer to this measure as the *mutual information* between X and Y denoted $I(X;Y)$, that intuitively quantifies the difference between the uncertainty $H(X)$ in X and the remaining uncertainty $H(X|Y)$ in X given knowledge about Y: (Cover and Thomas, 2006)

$$I(X;Y) = H(X) - H(X|Y) = \sum_{x \in \Omega_X, y \in \Omega_Y} p(x,y) log_2 \left(\frac{p(x,y)}{p(x)p(y)} \right)$$

Note that the sum in the previous equation is symmetric in X and Y. Indeed, $I(X;Y) = I(Y;X)$, whence the name *mutual* information.

In Figures 1.8 and 1.9 we provide examples of mutual information computed for different variables and a comparison between a linear analysis method such as correlation and mutual information which can capture nonlinear interactions.

Similarly, we define the mutual information between two variables X and Y conditioned on a third variable Z. It is referred to as *conditional mutual information* and is given by: (Cover

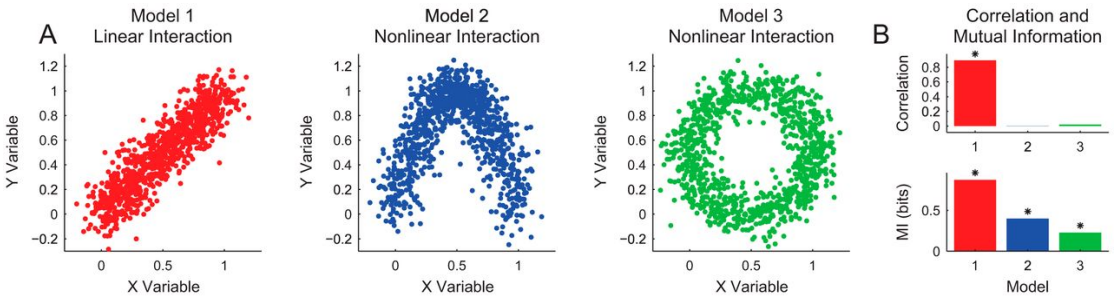


Figure 1.9 – Example of linear versus nonlinear analysis methods. A. Data for three models (red, blue, and green) with linear (red) and nonlinear (blue and green) interactions; B. the associated correlation coefficient and mutual information (MI) values for all three models. Figure 5 from (Timme and Lapish, 2018)

and Thomas, 2006)

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z) = \sum_{\substack{x\in\Omega_x, y\in\Omega_Y \\ Z\in\Omega_Z}} p(x,y|z)\, log_2\left(\frac{p(x,y|z)}{p(x|z)\,p(y|z)}\right)$$

Additionally, we can extend the definition of mutual information to groups of variables. e.g. for three variables X_1, X_2, and Y, we can define the mutual information between the pair (X_1, X_2) and Y as: (Cover and Thomas, 2006)

$$I(\{X_1, X_2\}; Y) = \sum_{\substack{x_1\in\Omega_{x_1}, x_1\in\Omega_{X_2} \\ y\in\Omega_y}} p(x_1,x_2,y)\, log_2\left(\frac{p(x_1,x_2,y)}{p(x_1,x_2)\,p(y)}\right)$$

This is a very helpful tool in neuroscience applications. For example, one could use the latter formula to compute how much information two neurons can provide together about a stimulus instead of individually.

Finally, it is important to note that an information-theoretic analysis can identify the existence of potentially complex interactions between variables in a system, but it does not provide a characterization of such interactions. Indeed, the result obtained when applying an information-theoretic measure is a number (in bits) that quantifies some relationship within the data. Surely, some intuition about how information is computed and what those numbers in bits mean can lead us to formulate hypotheses about interactions, however, it cannot directly help us explain those interactions. For example, when we said earlier that we could compute how much information two neurons can provide together about a stimulus instead of individually, such measure cannot tell us how these two neurons are interacting together. Fortunately, Williams and Beer (2010) have introduced *partial information decomposition*, which is a way to decompose mutual information between groups of variables into several intuitive components.

1.3.3 Partial Information Decomposition

When considering the mutual information $I(\{X_1, X_2\}; Y)$ between a pair of variables X_1 and X_2 and a third variable Y, the partial information decomposition allows us to decompose $I(\{X_1, X_2\}; Y)$ into four intuitive components: the *redundancy* R that is the information provided redundantly by both X_1 and X_2, the *unique information* provided by X_1 or X_2 alone, respectively U_1 and U_2, and the *synergy* S that is the extra information known about Y when X_1 and X_2 are known simultaneously. They satisfy the following equations.

$$I(\{X_1, X_2\}; Y) = S(X_1, X_2; Y) + R(X_1, X_2; Y) + U_1(X_1, X_2; Y) + U_2(X_1, X_2; Y)$$

$$I(X_1; Y) = R(X_1, X_2; Y) + U_1(X_1, X_2; Y)$$

$$I(X_2; Y) = R(X_1, X_2; Y) + U_2(X_1, X_2; Y)$$

Noticeably, these equations do not provide a unique definition for each of the partial components, in other words, one of the four components has to be defined first. Williams and Beer (2010) introduced the *minimum information* which can be interpreted as a measure of the redundancy. It is defined as:

$$I_{min}(\{X_1, X_2\}; Y) = \sum_{y \in \Omega_Y} \min_{\{X_1, X_2\}} \left[\sum_{x_i \in \Omega_{X_i}} p(x_i, y) log_2 \left(\frac{p(x_i, y)}{p(x_i) p(y)} \right) \right]$$

The partial information decomposition allows us to understand interactions between groups of neurons better and in section 2.2 it helps us detect complex interactions between neurons forming specific topological structures.

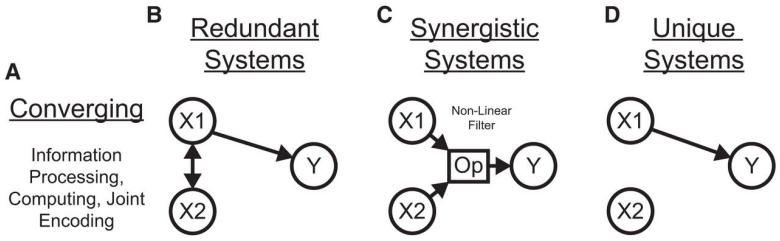


Figure 1.10 – A. Partial information interpretations in three example systems: B.Purely redundant system: the X variables provides the same amount of information about each state of Y. C. Purely synergistic system: the X variables alone provide no information about Y, however they do together via a nonlinear operation (Op). D. Purely unique system: Only X1 provides information about Y. Figure 7 from (Timme and Lapish, 2018)

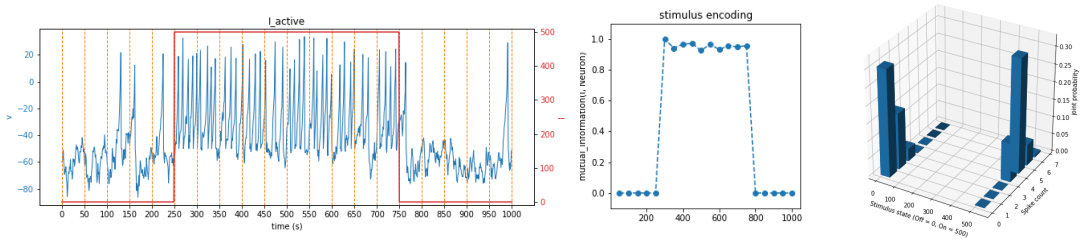
1.3.4 An Illustration: Stimulus Encoding by a Single Neuron

In the last part of this section, we present a very simple example inspired by (Timme and Lapish, 2018) of how one can apply information theory to data from neuroscience experiments.

This example represents a building block to our information-theoretic analysis in section 2.2.

The experiment consists of one square pulse applied to a single neuron. The data recorded consists of the number of spikes of the neuron in each time bin together with the state of the stimulus. Repeating the experiment multiple times, we can build a probability distribution from the data. We use this distribution to measure how the neuron encodes the stimulus, computed via the mutual information between the number of spikes of the neuron and the state of the stimulus.

In Figure 1.11a, we clearly observe a difference of spiking frequency of the neuron when the stimulus is active. This difference results in almost disjoint marginal distributions of the number of spikes of the neuron when the stimulus is inactive ($I = 0$) compared to when it is active ($I = 500$) (Figure 1.11c). Therefore, the mutual information between the two variables is very high, and can be translated into a high encoding of the stimulus by the neuron (Figure 1.11b).



(a) (b) (c)

Figure 1.11 – (a) Membrane potential of the neuron and the amplitude of the stimulus. (b) Stimulus encoding by the neuron.(c) Distribution of the number of spikes of the neuron.

2 Results

2.1 The Simple Framework of Choice

2.1.1 Motivation

Motivated by the combination of simplicity and power in both Izhikevich's neuron (Izhikevich, 2007) and the Brian simulator (Goodman and Brette, 2009), we take the initiative to build the *simple framework of choice*, an implementation of the simple model of choice using Brian. This framework provides a simple and powerful elementary tool to be used by scientists who wish to conduct a new kind of analysis (e.g. topological, information-theoric, ...) of neural network simulation.

2.1.2 Implementation

The implementation is straightforward thanks to the user-friendliness of Brian and the simplicity of Izhikevich's neuron. To implement a regular-spiking neuron, for example, one need only define the parameters of the model:

```
def dummy_neuron_of_choice():
    # Model's dimentionless parameters
    a = 0.03; b = -2; c = -50; d = 100
    # Neuron's biological parameters
    vr = -60; vt = -40; vpeak = 35; C = 100; k = 0.7
```

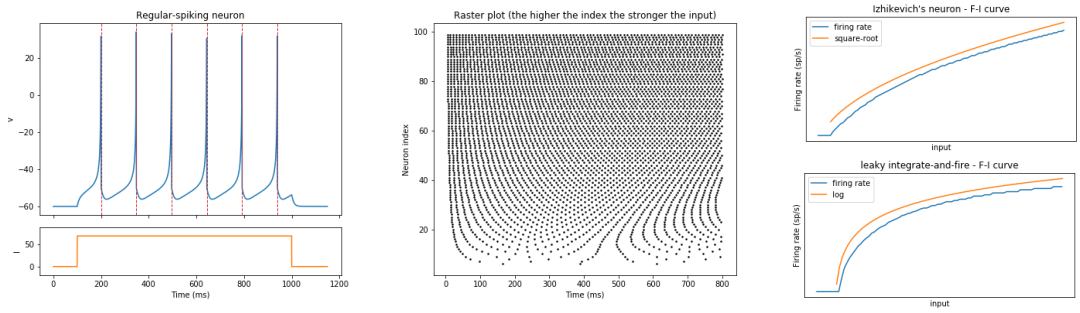
write the two-dimentional differential equation of Izhikevich's neuron and its reset condition:

```
    model = ''' dv/dt = (k*(v-vr)*(v-vt) - u)/(C*tau)  : 1
                du/dt = a*(b*(v - vr) - u)/tau : 1 '''
    reset = ''' v = c
                u += d '''
    peak_threshold = 'v>vpeak'
```

and give these variables as an input to the `NeuronGroup` Brian object:

```
    neuron = NeuronGroup(1, model, threshold=peak_threshold, reset=reset, method='euler')
```

In Figure 2.1a, we plot the membrane potential of this regular-spiking neuron when excited



(a) (b) (c)

Figure 2.1 – (a) Simple model receiving a constant input I. (b) Raster plot of neurons with increasing input strength. (c) Comparison of the F-I curve of the simple model versus the leaky integrate-and-fire model

with a constant input. In 2.1c we plot its firing rate in response to an increasing current input and compare it to the firing rate of the leaky integrate-and-fire. Consistent with what we said in the introduction, Izhikevich's model exhibits a square-root scaling while the integrate-and-fire model exhibits a logarithmic one. The full implementation of the model and the code to generate these plots can be found in A.1.

In what follows, we first use our implementation of the simple model to reproduce fundamental neurocomputational properties of biological neurons. Then, we run a large-scale simulation to demonstrate the benefits of our framework.

2.1.3 Reproducing Biological Neurocomputational Properties

In Figure 2.2, we reproduce 16 of the most fundamental neurocomputational properties of biological neurons by varying the four dimensionless parameters of the simple model of choice. Our framework is thus suitable for simulating a wide range of neural networks.
As an illustration, we use it in the next section (2.1.4) to simulate a network of 1000 neurons, similar to networks in mammalian cortexes, comprised of regular-spiking excitatory neurons and fast-spiking inhibitory neurons.

2.1.4 A Large-Scale Simulation

In this section, we reproduce the simulation used by Izhikevich (2003) to demonstrate the computational efficiency of his model. We use it as an example to show how one can use the framework and to demonstrate not only the computational efficiency of our framework but also its user-friendliness and handiness for neuroscientists.

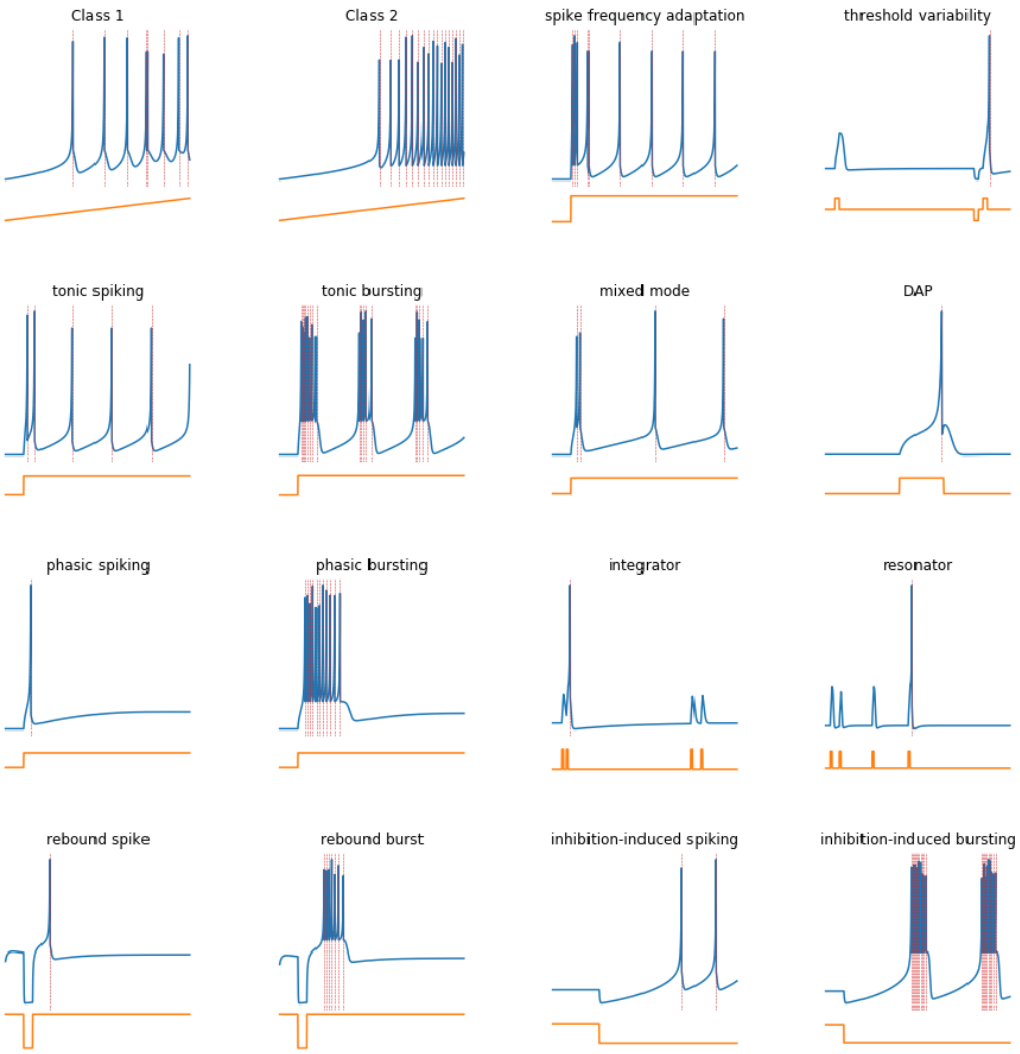
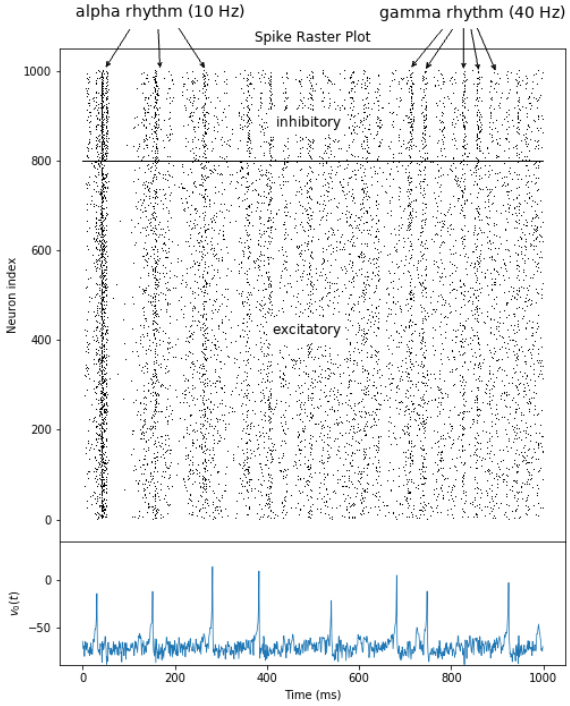


Figure 2.2 – The simple model used to reproduce 16 of the most fundamental neurocomputational properties of biological neurons. Inspired by Figure 8.8 from Izhikevich (2007).

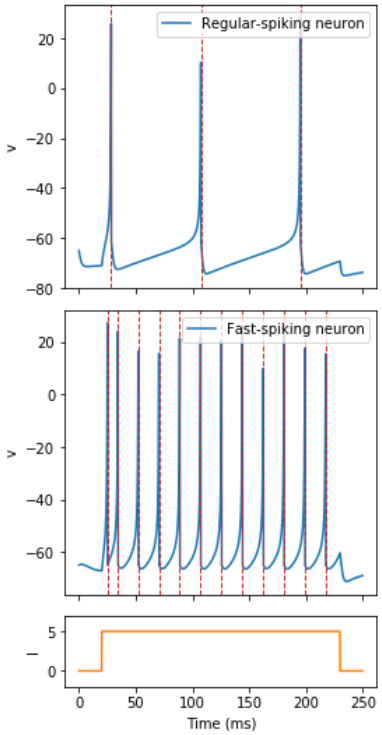
The network consists of 1000 neurons. 800 of which are modeled with a regular-spiking model and produce excitatory connections. The other 200 neurons are modeled with a fast-spiking model and produce inhibitory connections (Figure 2.3b). Randomization of the specific dynamics of each neuron is also applied to achieve heterogeneity. (Details of the simulation can be found in A.1).

Synapses are modeled with an instantaneous change in the membrane potential of the post-synaptic neuron. The value of this change is chosen randomly from a uniform distribution for each synapse. Inhibitory synapses are taken to be twice as strong as the excitatory ones. Additionally, each neuron receives a noisy input (applied directly to its membrane) following a normal distribution.

This setting is similar to the mammalian cortex in which the ratio of excitatory to inhibitory connection is 4 to 1 and inhibitory connections are stronger. Furthermore, the noisy input received by each neuron represents the noisy thalamic input.



(a) Simulation of a network of 1000 randomly coupled spiking neurons. Top: spike raster shows episodes of alpha and gamma band rhythms (vertical lines). Bottom: typical spiking activity of an excitatory neuron. Reproduced from (Izhikevich, 2003).



(b) Regular-spiking model vs fast-spiking model when stimulated with a constant input.

Figure 2.3

The simulation yields Figure 2.3a, in which we can observe cortical-like asynchronous dynamics: neurons seem to fire Poisson spike trains of mean firing rate equal to 8 Hz. Moreover, neurons occasionally fire synchronously at frequencies of 10 or 40 Hz. These synchronous firings are pictured by the dark vertical lines and represent alpha and gamma waves observed in the mammalian cortex in its awake state.

It is quite admirable that such a network with a fairly simple model of neurons, random connections, and no synaptic plasticity is able to reproduce complex biological dynamics. Indeed, Izhikevich is proud to say in his paper (Izhikevich, 2003) that thanks to his model, *there is no longer a contradiction between biological plausibility and computational efficiency of model neural networks.*
We would say that our framework makes this assertion even more true thanks to the power of the Brian package that, for instance, allows its users to define complex models of plastic

synapses while maintaining a computationally efficient implementation.

Furthermore, we wish to emphasize on the user-friendliness and handiness of our framework. To illustrate such characteristics, we compare the code we used to produce this simulation using our framework against the original code written by Izhikevich. We have translated the original MATLAB code to Python code for a more direct comparison.

Looking at the declaration of the neuron models and the synapses (Tables 2.1 and 2.2), one may have the first impression that the Brian code is clearly longer, however, by looking at the section of the code used to run the simulation (Table 2.3), one is happily surprised to find out that in the Brian code, the complex machinery needed track spikes and perform the numerical integration is abstracted away.

Plain Python (Izhikevich)

```
# Excitatory neurons
Ne = 800;
re = np.random.rand(Ne);
a = np.concatenate([0.02*np.ones(Ne),
b = np.concatenate([0.2*np.ones(Ne),
c = np.concatenate([-65+15*re**2,
d = np.concatenate([8-6*re**2,
```

```
Inhibitory neurons
Ni = 200
ri = np.random.rand(Ni)
0.02+0.08*ri])
0.25-0.05*ri])
-65*np.ones(Ni)])
2*np.ones(Ni)])
```

Simple Framework of Choice (Brian)

```
Ne = 800
excitatory_eq = '''
    re : 1
    a = 0.02 : 1 (shared)
    b = 0.2 : 1 (shared)
    c = -65 + 15*re**2 : 1
    d = 8 - 6*re**2 : 1

    dv/dt = (0.04*v**2 + 5*v + 140 - u)/tau
             + 5*sigma*xi*sqrt(1/tau)   : 1
    du/dt = a*(b*v - u)/tau : 1 '''

reset = ''' v = c
            u += d '''
vpeak = 30
peak_threshold = 'v>vpeak'

excitatory_group = NeuronGroup(
    Ne, excitatory_eq, threshold=peak_threshold,
    reset=reset, method='euler')
excitatory_group.re = 'rand()'

excitatory_spikemon =\
    SpikeMonitor(excitatory_group)
```

```
Ni = 200
inhibitory_eq = '''
    ri : 1
    a = 0.02 + 0.08*ri : 1
    b = 0.25 - 0.05*ri : 1
    c = -65 : 1 (shared)
    d = 2 : 1 (shared)

    dv/dt = (0.04*v**2 + 5*v + 140 - u)/tau
            + 2*sigma*xi*sqrt(1/tau) : 1
    du/dt = a*(b*v - u)/tau : 1 '''

reset = ''' v = c
            u += d '''
vpeak = 30
peak_threshold = 'v>vpeak'

inhibitory_group = NeuronGroup(
    Ni, inhibitory_eq, threshold=peak_threshold,
    reset=reset, method='euler')
inhibitory_group.ri = 'rand()'

inhibitory_spikemon =\
    SpikeMonitor(inhibitory_group)
```

Table 2.1 – Declaration of the neuron models

The Brian code is indeed longer, but it is more readable and quicker to write. These are the benefits of using the Brian package in our framework.
More precisely, in our framework the forward Euler integration is abstracted away via the argument `method='euler'` in the definition of the `NeuronGroup`. On the contrary, the declaration of the neuron groups and synapses is explicit: instead of concatenating the values of the parameters of the inhibitory neurons after the excitatory ones and making sure they

Plain Python (Izhikevich)

```
# Excitatory neurons                                Inhibitory neurons
S = np.concatenate([0.5*np.random.rand(Ne+Ni,Ne),  -np.random.rand(Ne+Ni,Ni)], axis=1)
```

Simple Framework of Choice (Brian)

```
S_EtoE = Synapses(excitatory_group, excitatory_group, 'w : 1', on_pre='v_post += 0.5*w')
S_EtoE.connect()
S_EtoE.w = 'rand()'

S_EtoI = Synapses(excitatory_group, inhibitory_group, 'w : 1', on_pre='v_post += 0.5*w')
S_EtoI.connect()
S_EtoI.w = 'rand()'

S_ItoI = Synapses(inhibitory_group, inhibitory_group, 'w : 1', on_pre='v_post -= 1*w')
S_ItoI.connect()
S_ItoI.w = 'rand()'

S_ItoE = Synapses(inhibitory_group, excitatory_group, 'w : 1', on_pre='v_post -= 1*w')
S_ItoE.connect()
S_ItoE.w = 'rand()'
```

Table 2.2 – Declaration of the synapses

Plain Python (Izhikevich)

```
for t in range(1,1000):                                                    # simulation of 1000 ms
    I = np.concatenate([5*np.random.randn(Ne), 2*np.random.randn(Ni)])     # thalamic input
    fired = np.nonzero(v >= 30)[0]                                         # indices of spikes
    firings.extend(fired)
    firing_times.extend(t + 0*fired)
    v[fired] = c[fired]
    u[fired] = u[fired] + d[fired]
    I = I + np.sum(S[:,fired], axis=1)
    v = v + 0.5*(0.04*v**2 + 5*v + 140 - u + I)    # step 0.5 ms
    v = v + 0.5*(0.04*v**2 + 5*v + 140 - u + I)    # for numerical
    u = u + a*(b*v - u)                            # stability
```

Simple Framework of Choice (Brian)

```
run(1000*ms)
```

Table 2.3 – Code needed to run the simulation

are aligned for each parameter (Plain Python), a user of our framework can define these parameters independently for each type of neuron. In addition, the declaration of the synapses is unfolded and divided into categories. Most importantly, the events of spiking, resetting membrane potentials, or activating synapses are each defined, handled, and recorded by their respective Brian object. This liberates scientists from implementing a loop and in which they have to write the code to approximate the evolution of the simulation and the code to find and record its important events. As the simulation gets more complex, with more neuron types and more intricate connectivity, this code becomes more and more prone to bugs and harder to write.

Indeed, the average scientist wishing to simulate a neural network should not be concerned about learning the intricacies of computer simulations. First, because they should spend most of their time on the details of their network rather than the details of its implementation. And second, because the Brian simulator will produce a computationally more efficient version

anyway.

Trained as a computer scientists to write *good* code, I have noticed that most of the code that researchers in neuroscience write to produce their simulations and plot their figures is very often lost forever and not reused. If not, it is for instance uploaded on GitHub with a mention "Lacks documentation. Provided for transparency and reproducibility". We hope that our framework will, in accordance with the spirit of Brian, help scientists more easily write code for simulating neural networks, encourage them to share this code with their community, and facilitate its review by other scientists.

2.2 Cliques of Neurons Create and Transform Information

2.2.1 Motivation

In (Reimann et al., 2017), the authors investigate how the local topological structure organizes the activity of neurons. They examine spike correlations between pairs of neurons within individual cliques and study how these correlations vary as the number of neurons forming the clique increases (Figure 2.4). They found that all correlations increased with simplex dimension and that some pairs, characterized by their position in the simplex when ordered from source to sink, were consistently more correlated than the others. Precisely, they focus on the pair consisting of the first (source) and last (sink) neurons (2.4:purple) which share the highest number of indirect connections, the pair consisting of the last (sink) and second-to-last neurons (2.4:red) which have the highest number of shared inputs, and the pair consisting of the first and the second neurons (2.4:green) which have the lowest number of shared inputs and indirect connections. The pair with the last two neurons in the simplex was the one scoring the highest correlation. This suggests that shared input generates more of the pairwise correlation in spiking than indirect connections in directed simplices.

In the section, we aim to better understand what makes this difference in correlation and what it can tell us about the role of local connectivity in shaping neural activity. To do so, we build a network consisting of an isolated clique of neurons and replace correlation as our studied measure by mutual information. We describe the details of our method in 2.2.2 and present our results in 2.2.3 and 2.2.4.

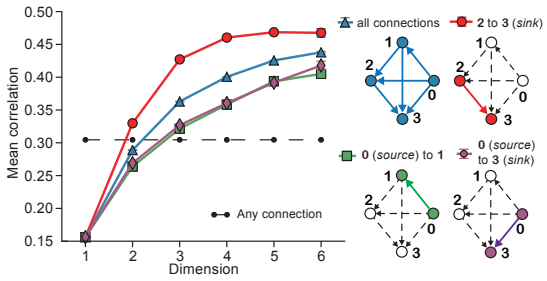


Figure 2.4 – Mean correlation coefficient of pairs of neurons, given their position within a simplex and its dimension. Figure 4.E from (Reimann et al., 2017)

2.2.2 Implementation: Isolating Cliques of Neurons in a Network

Using our simple framework (Section 2.1), we consider networks of n Izhikevich regular spiking neurons with parameters summarized in Table 2.4. The neurons receive a noisy input following a random distribution (making them fire about 0.8 times in 50 ms). We connect all the neurons together to form a directed clique, or precisely, an $n-1$-simplex. All synapses are excitatory and are modeled by a constant depolarization of magnitude w in the postsynaptic neuron membrane. In addition, we apply a stimulus consisting of a square pulse of amplitude

500 pA to only the source of the simplex. We simulate this network during 1000 ms, performing N_{trials} trials with the stimulus On between $t = 250$ ms and $t = 750$ ms and Off the rest of the time and N_{trials} trials with the stimulus always Off.

For each trial, we count the number of spikes of each neuron in time intervals of size t_{bin} ms and construct poststimulus time histograms (PSTHs) for each neuron. We use these histograms to build a joint probability distribution for each time bin describing the state of the stimulus and the number of spikes of each neuron and we use these distributions to conduct our information-theoretic analysis. Precisely, the distribution at bin_i gives the probability of the event $(y, x_{0,k}, x_{1,k}, \ldots, x_{n-1,k})$ happening, where y is the state of the stimulus (On, Off) and $x_{i,k}$ denotes the number of spikes of neuron i in the time bin k. The majority of the measures we used were computed with the python package `dit` (G. James et al., 2018).

By default we take $w = 10$ mV which makes a neuron fire when it receives about 2 or 3 excitatory postsynaptic potentials (EPSP) and $t_{\text{bin}} = 50$ ms. For this pair of values we can fit a full trail of spikes by all the neurons in the clique in one time bin (Figure 2.5). Moreover, we take $N_{\text{trials}} = 500$ which provides sufficiently accurate distributions. For additional control we also consider $w \in \{5, 7, 12, 15, 20, 25\}$and $t_{\text{bin}} \in \{10, 25, 100\}$ (See A.2).

The code to generate the observations can be found in A.2.

Parameter	Value
C	100
v_r	-60
v_t	-40
v_{peak}	35
k	0.7
a	0.03
b	-2
c	-50
d	100

Table 2.4 – Neuron model parameters

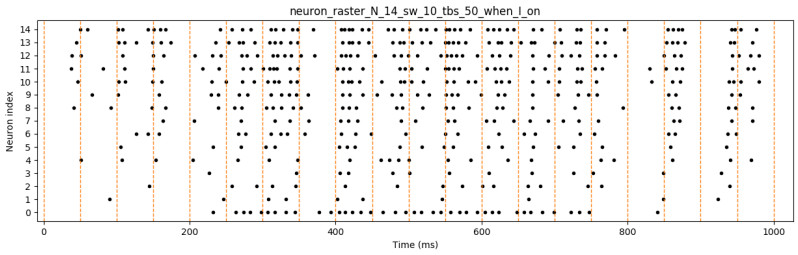
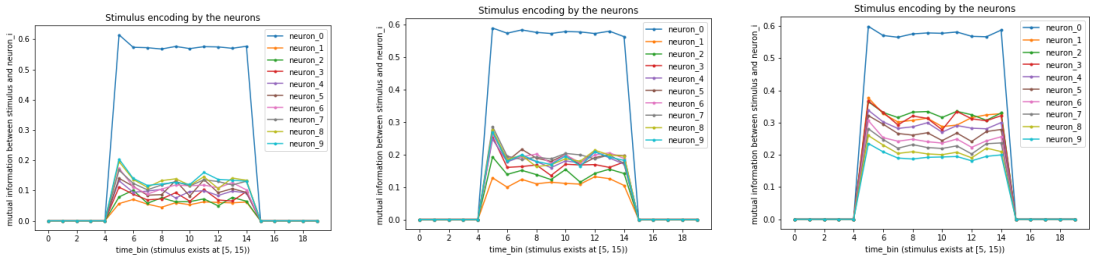


Figure 2.5 – Raster plot of a clique of 15 neurons for $w = 10, t_{bin} = 50ms$

2.2.3 Cliques Create Information

We first examine the encoding of the stimulus by each of the neurons 2.6. To measure it, we compute the mutual information between the state of stimulus and the number of spikes per time bin of each neuron. We observe that the first neuron (source) almost perfectly encodes the stimulus. This is expected since it receives the stimulus directly. The other neurons also encode the stimulus, though to a lesser degree. These values increase with w (Figure 2.6) and if we were to rank the level of encoding by the neurons according to their position in the simplex (except the source), it would be decreasing (last neuron encodes best) for low values

of w and increasing (second neuron encodes best) for high values of w. Indeed, for low values of w, a single synapse is not strong enough to communicate the stimulus directly. Thus, the last neurons that receive the most input encode the stimulus best. For high values of w, a single synapse is enough to transmit the stimulus and more synaptic input actually creates noise, which results in a better stimulus encoding for neurons closer to the source.



(a) $w = 10$ (b) $w = 15$ (c) $w = 30$

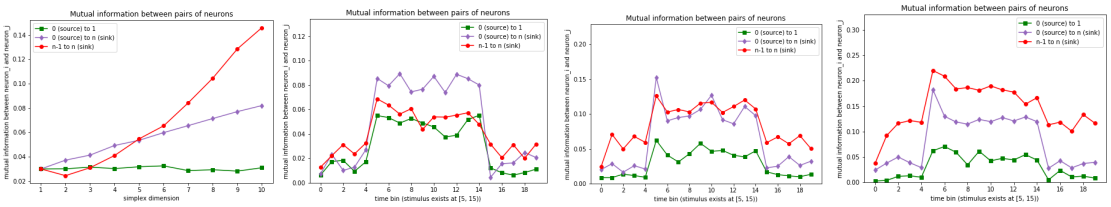
Figure 2.6 – Stimulus encoding in a clique of 10 neurons for various values of w

Second, inspired by Reimann et al. (2017), we examine the mutual information between the same pairs of neurons that the authors studied, namely, the pairs consisting of the first (source) and the second neurons (2.4:green), the first and the last (sink) neurons (2.4:purple), and the second-to-last and the last neurons (2.4:red). We obtain a rather different plot (Figure 2.7) from the one in (Reimann et al., 2017) shown in (Figure 2.4).

The first difference is that the mutual information is not increasing for the three pairs of neurons. Indeed, the pair consisting of the first (source) and second neuron exhibits a constant mutual information as the dimension of the simplex increases (green curve). This is consistent with our network since no change happens at the level of the first and second neurons as the dimension of the simplex increases. Actually, since in (Reimann et al., 2017) the correlation between this pair of neurons is increasing, the former argument suggests that simplices of higher dimensions are involved in a higher organized activity.

The second difference is that the last neuron (sink) shares a mutual information that is higher with the first neuron (source) than with second-to-last neuron for simplices of dimension less than 5 (purple curve vs red curve). The first key point that can help us justify this difference, is that the cliques we study in our network are isolated. That is, although some noise is applied to each neuron, the only input affecting the network is the stimulus directly applied to only the first neuron. Hence, at first glance, one can conjecture that the only information that can be shared between a pair of neurons must be information about the stimulus. Therefore, since the two neurons that best encode the stimulus are the source and the sink, this pair of neurons should have the highest mutual information. This may indeed hold for dimensions lower than 5, however, as the dimension of the simplex increases beyond 5, the mutual information between the sink and the neuron before the sink becomes significantly greater than that between the source and the sink. To understand this intersection, we look at the

mutual information between the different pairs at different time bins (Figure 2.7 b, c, and d). We observe that in dimension 4 (b), the mutual information between the source and the sink is significantly higher in the bins 5 to 14 during which the stimulus is playing a role. This supports our conjecture that the stimulus is the main source of difference in mutual information. However, in dimension 7 (b), the red and purple curves are almost equal when the stimulus is active and that the two curves differ in bins 0 to 4 and 15 to 19, that is, when the stimulus is always off. Eventually, by dimension 10, the mutual information between the last and second-to-last neurons becomes greater than that of the other pairs in all the time bins. These observations suggest information is somehow generated between the final neurons in high dimensional simplices and that this information eventually outweighs that coming from the stimulus, leading us to reject our previous conjecture. A potential candidate for this information is the information about the growing organization between the last neurons.



(a) Average mutual information in different dimensions (b) Simplex of dimension 4 (c) Simplex of dimension 7 (d) Simplex of dimension 10

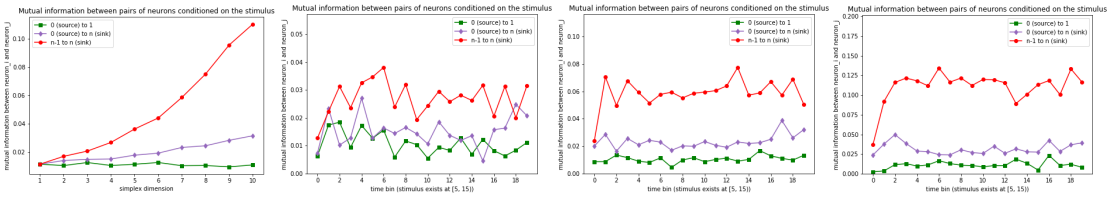
Figure 2.7 – Mutual information between pairs of neurons in a simplex

Furthermore, we can test this new hypothesis by computing the mutual information between the pairs of neurons conditioned on the state of stimulus, thus eliminating the effects of the stimulus in our analysis. We obtain (Figure 2.8 (a)) in which the red curve is always above the purple curve, supporting the hypothesis that the information shared by the last two neurons in the simplex does not come from the stimulus. In addition, we can observe that the three curves are almost constant during all the time bins for different dimensions (Figure 2.8 b, c, and d) confirming that the information from the stimulus is discarded. The plot in dimension 10, also shows an increase of the mutual information between the last two neurons from bin 0 to bin 2 illustrating the increase of the information generated by the simplex at the beginning of the simulation.

We have thus come closer to the conclusion that new information in created within high dimensional cliques of neurons that could reflect their high degree of organization.

2.2.4 Cliques Transform Information

Building upon the hypothesis developed in the previous section which states that new information is created within high dimensional cliques of neurons, we aim to investigate how this information is generated.



(a) Average conditional mutual information in different dimensions

(b) Simplex of dimension 4

(c) Simplex of dimension 7

(d) Simplex of dimension 10

Figure 2.8 – Mutual information between pairs of neurons in a simplex conditioned on the stimulus state

To do so, we apply the partial information decomposition framework described in 1.3.3. Precisely, we now consider the mutual information between the couple (stimulus, first element of a pair) and the second element of the pair. We then decompose the information coming from the couple (stimulus, first element of a pair) into four distinct components forming the partial information decomposition: The redundant information, the unique information coming from the stimulus, the unique information coming from the first element of the pair of neurons, and the synergistic information generated by the couple. As a measure of the redundancy, we use the minimum information (I_{min}).

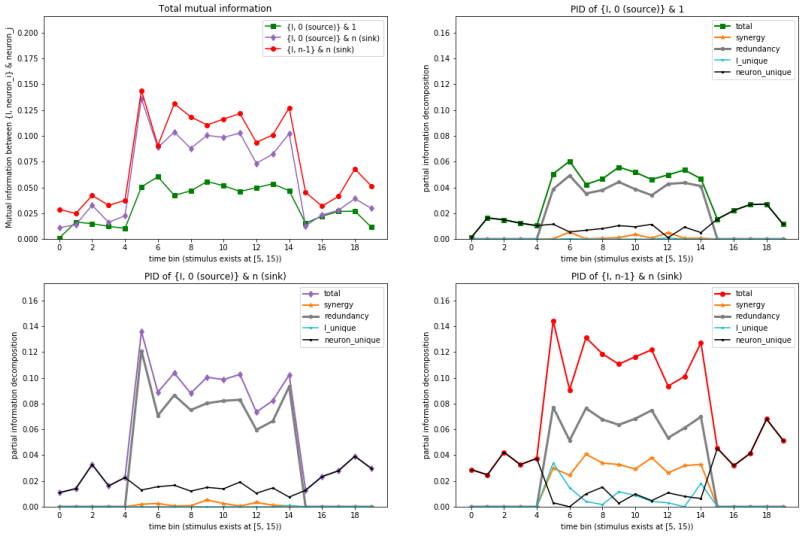


Figure 2.9 – Partial information decomposition applied in a clique of 7 neurons

In Figure 2.9, we plot the partial information decomposition of the three pairs of neurons we previously studied. We can observe that in both pairs in which the source neuron is involved (top right and bottom left), most of the information provided by the stimulus and the source is redundant (gray curve) and the rest of it comes uniquely from the source. This is not surprising,

the source neuron is transmitting the information it encodes from the stimulus. Hence most of it is redundant and a small part of it comes from the spiking properties of the neuron. On the contrary, in the pair consisting of the sink and the neuron before the sink plus the stimulus, we observe a totally different decomposition. Firstly, in bins 0 to 4 and 15 to 19, the information unsurprisingly comes uniquely form the second-to-last neuron (black curve). Secondly, and most remarkably, at bins 5 to 14, when the stimulus is involved, we can observe that there is no information that comes uniquely from the neuron anymore and that a significant amount of synergistic information generated by both the stimulus and the second-to-last neuron together appears. This is what we suggest to be a transformation of the input stimulus by the clique of neurons.

We then examine independently the evolution of each of the partial information components when the dimension of the clique increases (Figure 2.10). We observe again that the pairs involving the source neuron mostly generate redundant information and thus provide no interesting insights. However, the components of the information shared between the sink and the neuron before the sink plus the stimulus exhibit some interesting behaviors (red curves). The unique information coming from the stimulus starts high and then decreases. The synergistic information increases and then decreases. And the unique information coming from the second-to-last neuron starts low and then increases. Summing these three behaviors together, we can more clearly suggest that the information provided by the couple (stimulus, second-to-last neuron) starts as a unique information coming from the stimulus, then is transformed into synergistic information, and then becomes uniquely provided by the second-to-last neuron.

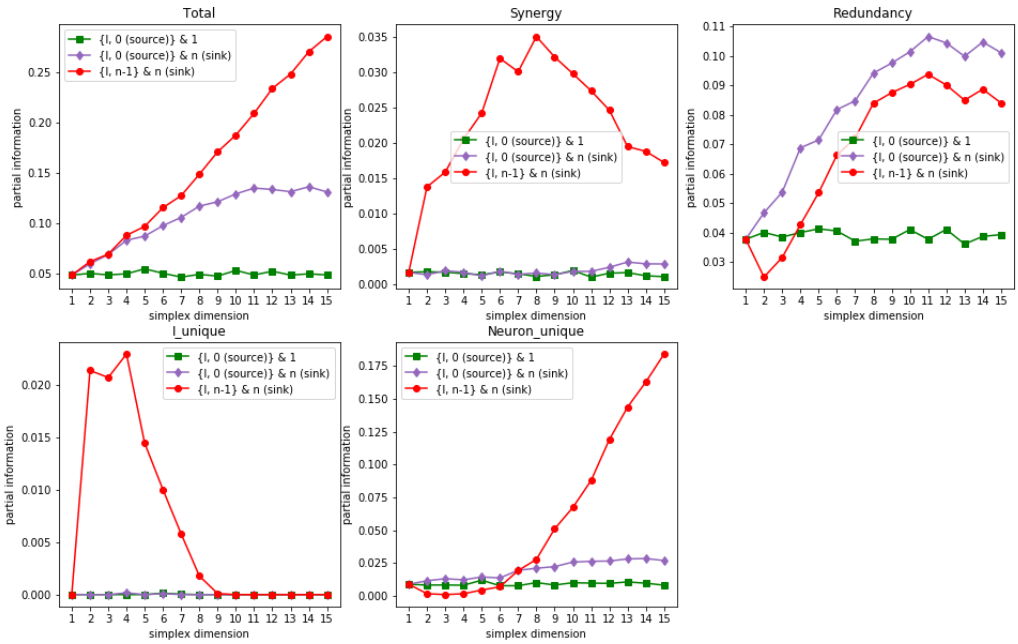


Figure 2.10 – Partial information decomposition by component in different dimensions

Taking another perspective on this result, we plot the four components together for each pair of neurons (Figure 2.11). We observe more clearly in the rightmost figure, corresponding to the last neurons in the clique, an increase of the unique information coming from the stimulus, followed by its decrease and an increase of the synergistic information, followed by its decrease and an increase in the information coming uniquely from the second-to-last neuron, further strengthening our "transformation of information" hypothesis. Moreover, in very high dimensions, we observe a decrease in the redundant information, and an audacious claim would be to suggest that the unique information coming from the second-to-last neuron could actually disguise a further transformation of the information occurring in the last neurons of the clique. An interesting analysis would be to consider the partial information decomposition of successive triplets of neurons in the clique.

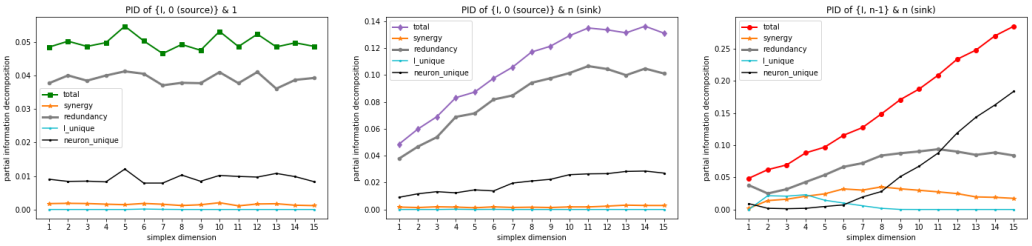


Figure 2.11 – Partial information decomposition for each pair in different dimensions

3 Conclusion

3.1 Summary of Main Results

The main goal of our research was to better understand how local topological structures in a neural network, such as directed cliques of neurons, shape its emergent dynamics. This was motivated by (Reimann et al., 2017), in which the authors demonstrate the presence of a remarkably high number and variety of high-dimensional directed cliques of neurons in digital reconstructions of neocortical microcircuitry and formulate a correlation between these topological structures and the function of neural networks.

To tackle this question, we first went on an exploration of the multiple disciplines that combined would help us realize our goal. We described the basics of neuroscience from a computational point of view, introduced the concepts of algebraic topology that are currently contributing to significant progress in the field of neuroscience, and described several tools from information theory that would make the novelty of our research.

This multidisciplinary exploration made us realize the necessity of building our own framework to perform the simulations we would need to study our question of interest. Consequently, as a second step, in section 2.1, we described the *simple framework of choice*, which was the framework we succeeded to set up combining Izhikechich's simple neuron model and the simulator Brian. Bringing together the benefits of both of these tools, our framework is user-friendly, computationally efficient and accurate, and easy to review. These are the three desired characteristics we aimed for to properly introduce our novel information-theoretic analysis.

Thirdly, leveraging our framework and the tools from information theory we described in our introduction, we were able to formulate new conclusions about the dynamics of neurons forming directed cliques. We showed that when subject to a stimulus, high dimensional cliques transform the information coming from the stimulus to generate new information reflecting their high degree of organization. Decomposing this information into four disjoint partial components, we further found that the information generated by the last neurons of

the clique is created synergistically with the information coming from the stimulus, whence our hypothesis that cliques of neurons transform information coming from a stimulus.

3.2 Discussion and Future Directions

Generalizing the results described in 2.2 to more general settings would be undoubtedly premature. Our current results only apply to the specific neural network model we considered. In particular, we only considered one type of neuron, the regular spiking Izhikevich neuron, and a fairly simple model of synapses: a constant depolarization of the postsynaptic neuron. Furthermore, the minimum information I_{min} we used for the partial information decomposition in 2.2.4 is only one candidate among many others. We chose it because it was the first measure introduced by Williams and Beer (2010) and because it possesses several advantages over the other measures such as its intuitive definition and its possible extension to many variables. Nevertheless, the minimum information has also been shown to measure the redundant amount of information rather than the redundant content, which Harder et al. (2013) consider not to be a fully satisfactory way to capture the concept of redundancy.

In this regard, we believe that the following directions should be explored to complement and expand on our results. First, one can test different measures of redundancy and synergy in the partial information decomposition. Then, one can study the mutual information between more than pairs of neurons in a clique to further examine how the synergistic information evolves in the clique. Generalizing the previous idea, one can study the mutual information between cliques which are faces of bigger cliques to have a broader idea on the interactions between the local topological structures. Additionally, one could consider global topological structures and examine the mutual information between cliques forming cavities. As a second step, one could explore all of the previously described measures of information in simulations of small biological neural networks such as the *C. elegans* connectome which has been extensively studied. Finally, it would interesting to apply this novel information-theoretic analysis to the digital reconstructions of neocortical microcircuitry built by the Blue Brain project.

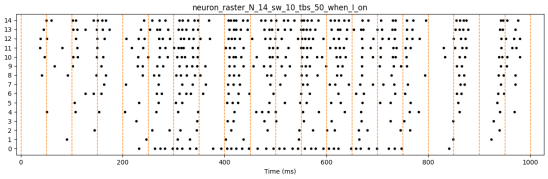
A Appendix

A.1 The Simple Framework of choice

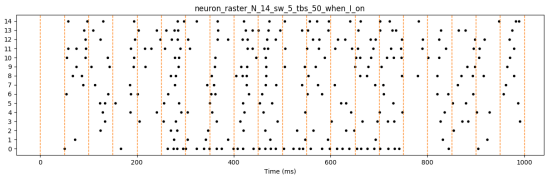
https://github.com/skandermoalla/simple-framework-of-choice.

A.2 Cliques of Neurons Create and Transform Information

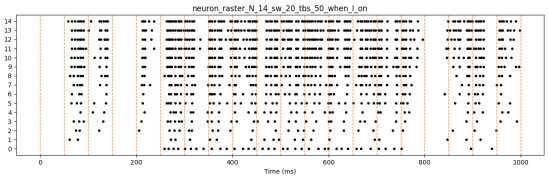
https://github.com/skandermoalla/it-simplex



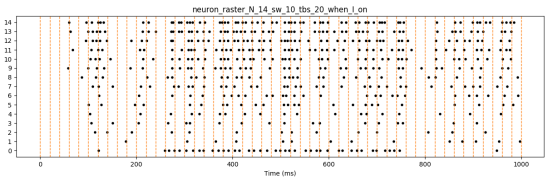
(a) $t_{bin} = 50, w = 10$



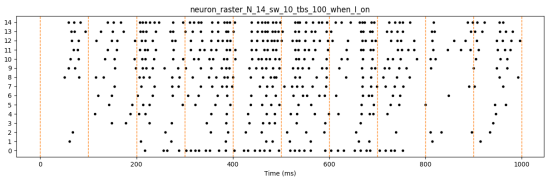
(b) $t_{bin} = 50, w = 5$



(c) $t_{bin} = 50, w = 20$

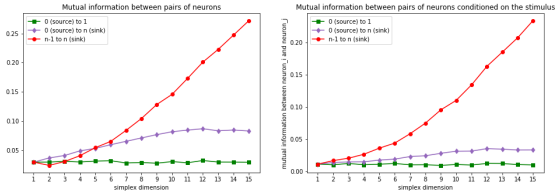


(d) $t_{bin} = 20, w = 10$

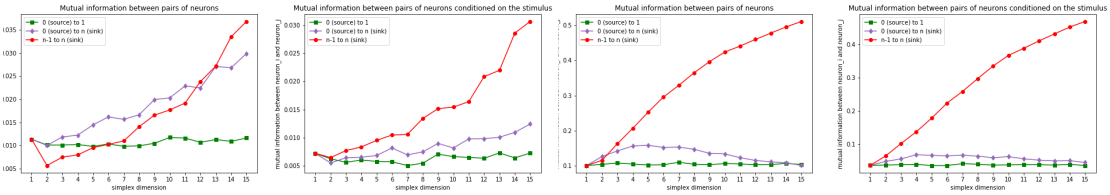


(e) $t_{bin} = 100, w = 10$

Figure A.1 – Neuron rasters for different values of w and t_{bin}

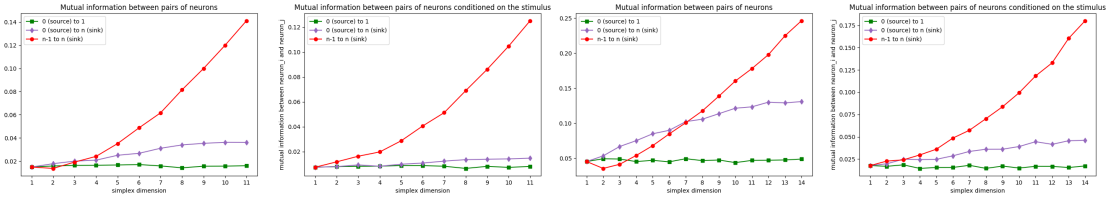


(a) $t_{bin} = 50, w = 10$



(b) $t_{bin} = 50, w = 5$

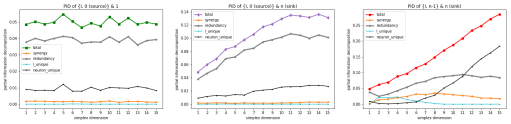
(c) $t_{bin} = 50, w = 20$



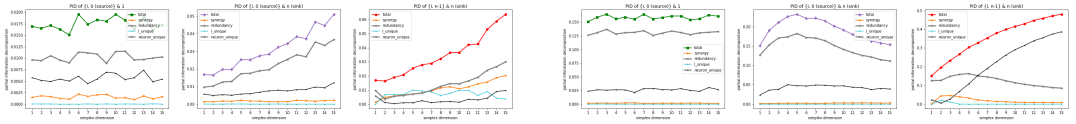
(d) $t_{bin} = 20, w = 10$

(e) $t_{bin} = 100, w = 10$

Figure A.2 – Mutual information between pairs of neurons for different values of w and t_{bin}

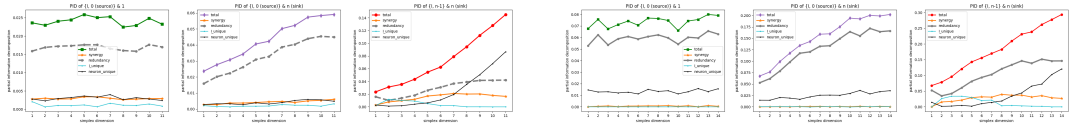


(a) $t_{bin} = 50, w = 10$



(b) $t_{bin} = 50, w = 5$

(c) $t_{bin} = 50, w = 20$



(d) $t_{bin} = 20, w = 10$

(e) $t_{bin} = 100, w = 10$

Figure A.3 – Partial information decompositions for different values of w and t_{bin}

Bibliography

Bardin, J.-B., Spreemann, G., and Hess, K. (2019). Topological exploration of artificial neuronal network dynamics. *Network Neuroscience*, 3(3):725–743. Publisher: MIT Press.

Cover, T. M. and Thomas, J. A. (2006). *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA.

Curto, C., Geneson, J., and Morrison, K. (2018). Fixed Points of Competitive Threshold-Linear Networks. *Neural Computation*, 31(1):94–155. Publisher: MIT Press.

Curto, C., Langdon, C., and Morrison, K. (2019). Robust motifs of threshold-linear networks. *arXiv:1902.10270 [q-bio]*. arXiv: 1902.10270.

G. James, R., J. Ellison, C., and P. Crutchfield, J. (2018). dit: a Python package for discrete information theory. *Journal of Open Source Software*, 3(25):738.

Garin, A. and Tauzin, G. (2019). A Topological "Reading" Lesson: Classification of MNIST using TDA. *arXiv:1910.08345 [cs, math, stat]*. arXiv: 1910.08345.

Goodman, D. F. M. and Brette, R. (2009). The Brian simulator. *Frontiers in Neuroscience*, 3. Publisher: Frontiers.

Harder, M., Salge, C., and Polani, D. (2013). Bivariate measure of redundant information. *Physical Review E*, 87(1):012130. Publisher: American Physical Society.

Hatcher, A. (2002). *Algebraic Topology*. Cambridge University Press. Google-Books-ID: BjKs86kosqgC.

Hodgkin, A. L. and Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, 117(4):500–544.

Izhikevich, E. (2003). Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6):1569–1572. Conference Name: IEEE Transactions on Neural Networks.

Izhikevich, E. M. (2007). *Dynamical Systems in Neuroscience*. MIT Press.

Kanari, L., Dłotko, P., Scolamiero, M., Levi, R., Shillcock, J., Hess, K., and Markram, H. (2018). A Topological Representation of Branching Neuronal Morphologies. *Neuroinformatics*, 16(1):3–13.

Kanari, L., Ramaswamy, S., Shi, Y., Morand, S., Meystre, J., Perin, R., Abdellah, M., Wang, Y., Hess, K., and Markram, H. (2019). Objective Morphological Classification of Neocortical Pyramidal Cells. *Cerebral Cortex*, 29(4):1719–1735. Publisher: Oxford Academic.

Latora, V. and Marchiori, M. (2001). Efficient Behavior of Small-World Networks. *Physical Review Letters*, 87(19):198701. Publisher: American Physical Society.

Li, W. and Li, Y. (2019). Entropy, mutual information, and systematic measures of structured spiking neural networks. *arXiv:1912.01507 [math, q-bio]*. arXiv: 1912.01507.

Luetgehetmann, D., Govc, D., Smith, J., and Levi, R. (2019). Computing persistent homology of directed flag complexes. *arXiv:1906.10458 [math, q-bio]*. arXiv: 1906.10458.

Markram, H., Muller, E., Ramaswamy, S., Reimann, M. W., Abdellah, M., Sanchez, C. A., Ailamaki, A., Alonso-Nanclares, L., Antille, N., Arsever, S., Kahou, G. A. A., Berger, T. K., Bilgili, A., Buncic, N., Chalimourda, A., Chindemi, G., Courcol, J.-D., Delalondre, F., Delattre, V., Druckmann, S., Dumusc, R., Dynes, J., Eilemann, S., Gal, E., Gevaert, M. E., Ghobril, J.-P., Gidon, A., Graham, J. W., Gupta, A., Haenel, V., Hay, E., Heinis, T., Hernando, J. B., Hines, M., Kanari, L., Keller, D., Kenyon, J., Khazen, G., Kim, Y., King, J. G., Kisvarday, Z., Kumbhar, P., Lasserre, S., Le Bé, J.-V., Magalhães, B. R. C., Merchán-Pérez, A., Meystre, J., Morrice, B. R., Muller, J., Muñoz-Céspedes, A., Muralidhar, S., Muthurasa, K., Nachbaur, D., Newton, T. H., Nolte, M., Ovcharenko, A., Palacios, J., Pastor, L., Perin, R., Ranjan, R., Riachi, I., Rodríguez, J.-R., Riquelme, J. L., Rössert, C., Sfyrakis, K., Shi, Y., Shillcock, J. C., Silberberg, G., Silva, R., Tauheed, F., Telefont, M., Toledo-Rodriguez, M., Tränkler, T., Van Geit, W., Díaz, J. V., Walker, R., Wang, Y., Zaninetta, S. M., DeFelipe, J., Hill, S. L., Segev, I., and Schürmann, F. (2015). Reconstruction and Simulation of Neocortical Microcircuitry. *Cell*, 163(2):456–492.

Morrison, K. and Curto, C. (2019). Chapter 8 - Predicting Neural Network Dynamics via Graphical Analysis. In Robeva, R. and Macauley, M., editors, *Algebraic and Combinatorial Computational Biology*, MSE/Mathematics in Science and Engineering, pages 241–277. Academic Press.

Otter, N., Porter, M. A., Tillmann, U., Grindrod, P., and Harrington, H. A. (2017). A roadmap for the computation of persistent homology. *EPJ Data Science*, 6(1):17.

Perea, J. A. (2018). A Brief History of Persistence. *arXiv:1809.03624 [cs, math]*. arXiv: 1809.03624.

Reimann, M. W., Gevaert, M., Shi, Y., Lu, H., Markram, H., and Muller, E. (2019). A null model of the mouse whole-neocortex micro-connectome. *Nature Communications*, 10(1):3903.

Reimann, M. W., Nolte, M., Scolamiero, M., Turner, K., Perin, R., Chindemi, G., Dłotko, P., Levi, R., Hess, K., and Markram, H. (2017). Cliques of Neurons Bound into Cavities Provide a Missing Link between Structure and Function. *Frontiers in Computational Neuroscience*, 11.

Shannon, C. E. (1948). A Mathematical Theory of Communication. *Bell System Technical Journal*, 27(3):379–423. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/j.1538-7305.1948.tb01338.x.

Sizemore, A. E., Phillips-Cremins, J. E., Ghrist, R., and Bassett, D. S. (2018). The importance of the whole: Topological data analysis for the network neuroscientist. *Network Neuroscience*, 3(3):656–673.

Sporns, O. (2013). Structure and function of complex brain networks. *Dialogues in Clinical Neuroscience*, 15(3):247–262.

Stimberg, M., Brette, R., and Goodman, D. F. (2019). Brian 2, an intuitive and efficient neural simulator. *eLife*, 8:e47314. Publisher: eLife Sciences Publications, Ltd.

Timme, N. M. and Lapish, C. (2018). A Tutorial for Information Theory in Neuroscience. *eNeuro*, 5(3).

Topaz, C. M., Ziegelmeier, L., and Halverson, T. (2015). Topological Data Analysis of Biological Aggregation Models. *PLOS ONE*, 10(5):e0126383. Publisher: Public Library of Science.

Williams, P. L. and Beer, R. D. (2010). Nonnegative Decomposition of Multivariate Information. *arXiv:1004.2515 [math-ph, physics:physics, q-bio]*. arXiv: 1004.2515.