Dynamical effects of degree correlations in networks of type I model neurons

A dissertation presented in partial fulfilment of the
requirements for the degree of

Doctor of Philosophy
in
Mathematics

at Massey University, Auckland, New Zealand.

Christian Bläsche

2020

Abstract

The complex behaviour of human brains arises from the complex interconnection of the well-known building blocks – neurons. With novel imaging techniques it is possible to monitor firing patterns and link them to brain function or dysfunction. How the network structure affects neuronal activity is, however, poorly understood. In this thesis we study the effects of degree correlations in recurrent neuronal networks on self-sustained activity patterns.

Firstly, we focus on correlations between the in- and out-degrees of individual neurons. By using Theta Neurons and Ott/Antonsen theory, we can derive a set of coupled differential equations for the expected dynamics of neurons with equal in-degree. A Gaussian copula is used to introduce correlations between a neuron's in- and out-degree, and numerical bifurcation analysis is used determine the effects of these correlations on the network's dynamics. We find that positive correlations increase the mean firing rate, while negative correlations have the opposite effect.

Secondly, we turn to degree correlations between neurons – often referred to as degree assortativity – which describes the increased or decreased probability of connecting two neurons based on their in-or out-degrees, relative to what would be expected by chance. We present an alternative derivation of coarse-grained degree mean field equations utilising Theta Neurons and the Ott/Antonsen ansatz as well, but incorporate actual adjacency matrices. Families of degree connectivity matrices are parametrised by assortativity coefficients and subsequently reduced by singular value decomposition. Thus, we efficiently perform numerical bifurcation analysis on a set of coarse-grained equations. To our best knowledge, this is the first time a study examines the four possible types of degree assortativity separately, showing that two have no effect on the networks' dynamics, while the other two can have a significant effect.

Acknowledgement

First and foremost of all, I thank my supervisor Carlo Laing for his steady support throughout the entire time of my PhD. I greatly appreciate his most nonjudgemental and trusting character, as well as his knowledgeable and wise guidance.

Secondly, I thank Gaven Martin for being my co-supervisor and his comments on this work.

I thank Shawn Means for the valuable discussions. Further, Franziska Peter significantly improved this thesis with endless remarks on my manuscript – thanks a lot.

For their unparalleled love and support across the globe I thank my parents, my sister and my nephew.

I am very grateful for all the friends I made during this time in New Zealand without whom this work would not have been possible.

Thanks to my siblings in faith Dr. Friedri and Dr. Fränzi.

Contents

List of Figures

Chapter 1

Introduction

Despite extensive research on the human brain, it remains to a very large extent a rather mystical organ - even though the basic mechanisms are well understood. The challenging complexity arises from its size: A human brain is said to consist of an unimaginable amount of 100 billion nerve cells, or neurons, and well over 100 trillion directed interconnections. Together they form the brain atlas or connectome. Like any living cell, neurons are born and die. However, the synaptic connections between neurons are changeable within their lifetime. This structural framework allows for different dynamical patterns of neuronal activity over time, which in turn strengthens or weakens certain synaptic pathways, like water in a riverbed. Potentially, the connectome carries everything from one's memories, personality and behavioural characteristics to thought patterns and mental disorders [Seu12]. Novel imaging techniques and increasing computational power have led to ambitious, large-scale and long-term research projects mapping out an entire human connectome [EUA+12] and eventually simulating neurons on it [Mar06]. How the underlying structure of a neuronal network supports and interacts with emergent patterns of activity is an essential part in understanding the brain, and still remains a largely open question, which will be hard to answer with these simulations [JK17].

However, advances have been made. On a single cell level it has been possible to identify the function of particular neurons, and various connectomes have been mapped, ranging from the nervous system of C. elegans to rat brains. The investigation of neuronal networks often involves a large amount of neurons, and due to experimental and computational limits a feasible approach is to consider groups of neurons – called populations. Within the field of neural coding it is an open question if some sort of averaging is not already part of the information process. A lot of research has been conducted on a more coarse level. For example, the increased or decreased level of activity in the right middle occipital gyrus can be associated with disorders like depression and anxiety [SOSE+16]. Studies also show

that schizophrenia patients have an altered connectivity between functional regions of the brain when compared to healthy subjects [CSK+18], which could explain their auditory hallucinations. A person suffering from Parkinson's disease shows highly synchronised neural activity in the external globus pallidus (GPe), whereas in a healthy condition pairs of neurons fire in an almost uncorrelated way [SHZ+13]. A potential cause for the lost ability of the network to desynchronise might be through damage to the synaptic architecture due to an injury or adaptive remodelling. Are a neuron's internal biochemical properties entirely responsible for those activity patterns? Or can the network structure influence them as well? Does the brain exhibit heterogeneous structures on a cellular scale at all?

It turns out that brains are highly structured, containing densely and sparsely connected regions with selective coupling present [Spo10, dSSSNL+14]. Evidence for a causal connection between structural aspects and activity is given in [TIM+14]. Those authors show that the firing frequencies of cultured neurons from the cortex or hippocampus are correlated with their number of connections - the more connections the higher their firing frequency.

In the jargon of network theory neurons are referred to as *nodes* and synaptic connections as *links*. The number of links connecting to a node is the node's *degree*. The total count of nodes and links, degrees and their distribution are the most fundamental structural features. But network theory considers many others: shortest path length, clustering coefficient, closeness centrality or motif frequencies to name a few [RS10]. Rather basic and often discussed, yet not well explored in the context of neuronal networks are degree correlations. As synaptic connections exhibit an inherent direction of information flow, it is necessary to distinguish between the number of incoming and outgoing links, that is in- and out-degree respectively. Thus, we distinguish between two cases: First, degree correlations can occur within neurons, implying for a positive (negative) correlation that a neuron's in- and out-degree are rather similar (dissimilar). This type and its dynamical consequences will be of concern to us in Chapter 2. Second, we turn to degree correlation across links, which is commonly referred to as degree assortativity. Positive (negative) degree assortativity describes a preferred attachment between nodes sharing similar (dissimilar) degrees. There are four different kinds depending on whether in- or out-degree of the pre- and postsynaptic neuron are considered. In Chapter 3 we investigate those four different assortativity types.

Simulating neuronal networks is often computationally costly due to the large number of involved neurons. This issue is commonly addressed by averaging over groups or populations of neurons. In general, there are popular and proven ways of constructing such *mean field theories*. How can we incorporate all the relevant structural information? Is it possible to introduce those different correlations independently from another and to what extent? The second main focus of this thesis will be answering such questions.

The thesis is structured as follows:

The remainder of this Chapter gives an introduction to network theory and the relevant structural properties, followed by a short description of the electrochemical mechanisms of a neuron and how they have been utilised to build mathematical models. Subsequently, we look at how our model of choice, the theta neuron model, is embedded in the broad field of neuronal models.

The topic of Chapter 2 is to identify how neuronal dynamics change when positive or negative degree correlations within neurons are present in a network. We present the mean field model and show how degree correlations can be implemented in this theory. The next step is then the application of a dimension reduction technique. We present our findings before we compare them to results using more complex Morris-Lecar neurons.

For the degree assortativity study in Chapter 3 we derive similar mean field equations in a novel and intuitive way. We introduce a scheme to generate connectivity matrices with a desired degree assortativity of any kind, before we discuss further reduction techniques for an efficient implementation. As a result we find that (in,in) assortativity has major dynamical implications, (in,out) assortativity only minor and the remaining two, (out,out) and (out,in), do not affect neuronal dynamics at all.

Chapter 4's topic is an analysis of numerical methods we have used in previous chapters, ranging from singular value decomposition over numerical continuation to Gaussian quadrature and copulas. Chapter 5 is dedicated to the software module which has been developed for the numerical simulations of this thesis. We finish this thesis in Chapter 6 with a summary of our conclusions.

1.1 Networks

A network is a system of interconnected elements. The elements are commonly referred to as nodes or vertices and the connections between them as links or edges. A network described on an abstract mathematical level is called a graph and graph theory became a universal cross-disciplinary and powerful tool over recent decades [AB02]. In some cases it can be hard to apply the concept of a graph to a concrete problem. In neuroscience nodes represent neurons, groups of neurons, or brain regions and links represent the connections between them. A graph with N vertices and N_e edges is denoted by $G(N, N_e)$. In neuronal networks with synaptic coupling, edges are directed since neurons receive input from one group of neurons and transmit their action potentials to another group. When neurons mutually influence their membrane voltage due to their physical proximity, this is known as gap junction coupling. This can be modelled with undirected edges, but they will not be subject of this thesis. The adjacency matrix A fully describes the entire network connectivity. Its entry A_{ij} is equal to the number of edges from node j to node i, hence A is an $N \times N$

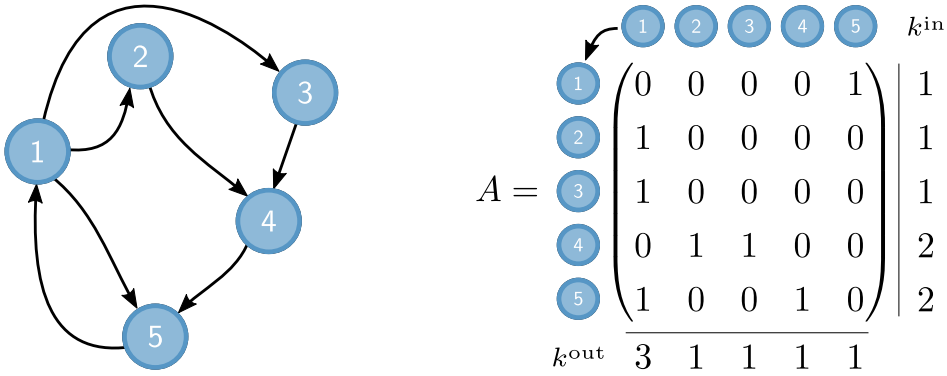

Figure 1.1: A simple, directed graph comprising 5 nodes and 7 edges (left) and its adjacency matrix A (right). We follow the convention that A_{ij} is the count of edges going from node j to node i. The summation over a row yields the in-degree, whereas the sum over a column is the out-degree.

matrix. A graph is said to be *simple* if there are neither multi-edges nor self-edges. This can easily be read from A. This is $A_{ij} \in \{0, 1\}; \forall i, j$ and $A_{ii} = 0; \forall i$. An example is given in Figure 1.1.

All structural properties of a network can be extracted from A. First of all, the number of nodes N is given by its number of columns or rows, since it is an $N \times N$ matrix, and the number of edges N_e equals the sum of all matrix entries

$$\sum_{i=1}^{N} \sum_{j=1}^{N} A_{ij} = N_e \tag{1.1.1}$$

A fundamental quantity of central relevance for this thesis is the node degree (or simply degree), which is the sum of a node's connections. In an undirected network, this is a single integer, but with directed edges we separately sum over incoming and outgoing connections to compute the in- and out-degree respectively. Those two values can be obtained from A by

$$k_i^{\text{in}} = \sum_{j=1}^{N} A_{ij} \qquad \text{and} \qquad k_i^{\text{out}} = \sum_{j=1}^{N} A_{ji} \tag{1.1.2}$$

Thus, we write a neuron i's degrees as the tuple $(k_i^{\text{in}}, k_i^{\text{out}}) = \boldsymbol{k}_i$. The sequence $(\boldsymbol{k}_i)_{i=1}^{N}$ is called the degree sequence and the set of all distinct degrees the degree space. The mean

degree of a network $\langle k \rangle$ can be computed from either the in- or out-degree sequence

$$\langle k \rangle = \frac{1}{N} \sum_{i=1}^{N} k_i^{\text{in}} = \frac{1}{N} \sum_{i=1}^{N} k_i^{\text{out}} = \frac{N_e}{N} \tag{1.1.3}$$

For the treatment of larger networks or networks where only statistical properties are known, it is valuable to introduce the notion of a degree distribution. When constructing adjacency matrices later on, we typically start with a given degree probability $P(\boldsymbol{k})$, which specifies the probability of finding the degree $\boldsymbol{k}$ when randomly picking a node. $P(\boldsymbol{k})$ is normalised with respect to the degree space by $\sum_{\boldsymbol{k}} P(\boldsymbol{k}) = 1$. Due to the difficulty of collecting connectivity data from real brains, a common approach is to make use of random networks. They are generated from certain statistical properties, such as the degree probability, to resemble their real counterparts. Typical degree distributions measured from brain regions scale with k^{-2} to k^{-3} [ECC+05]. In order to construct a network we first sample a degree sequence from $P(\boldsymbol{k})$. Subsequently, there are several methods to assemble an adjacency matrix A from this sequence – some are more probabilistic than others and thus the actual degree distribution of that particular network realisation may differ slightly from $P(\boldsymbol{k})$, thus

$$P(\boldsymbol{k}) \approx \frac{\sum_{i=1}^{N} \delta_{\boldsymbol{k}_i \boldsymbol{k}}}{N} \tag{1.1.4}$$

with $\delta_{\boldsymbol{k}_i \boldsymbol{k}}$ being the Kronecker delta.

1.1.1 Degree correlations

Along with the previously mentioned properties, degree correlations can be considered the most basic structural feature of a network. We distinguish between two types of correlations: firstly the correlation between in- and out-degree within each node and secondly between in- and out-degree across links. The latter is also called degree assortativity and actually splits up into four different cases. However, a study where the different types of correlations are investigated *separately* is missing from the literature. With this thesis we aim to contribute to filling this gap.

Degree correlation within nodes Correlations can occur between each node's in- and out-degree. This is between k^{in} and k^{out} in all tuples $\boldsymbol{k}$ (Figure 1.2). In this case, a positive (negative) correlation implies that neurons with many incoming connections have a large (small) number of outgoing connections. We use the Pearson correlation coefficient ρ as a

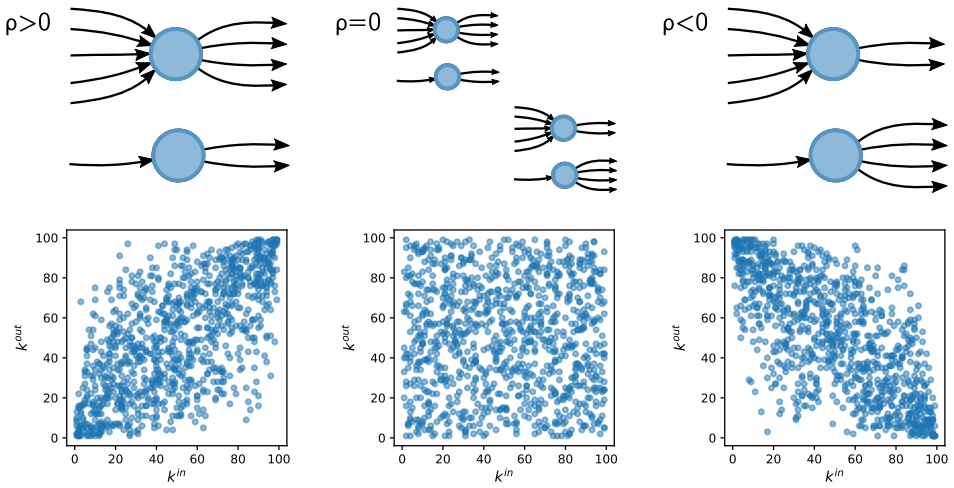

Figure 1.2: The three columns correspond to positive (left), neutral (centre), and negative (right) degree correlation within nodes. The top row illustrates in-/out-degree configurations which are favourable in the respective network, whereas the lower row shows the degree space with points for each node according to their in- and out-degree. In the case of neutral correlation ($\rho = 0$), chances for a node with high (or low) in-degree to have a high or low out-degree are equally likely.

measure of this property:

$$\rho = \frac{\sum_{i=1}^{N}(k_i^{\text{in}} - \langle k \rangle)(k_i^{\text{out}} - \langle k \rangle)}{\sqrt{\sum_{i=1}^{N}(k_i^{\text{in}} - \langle k \rangle)^2}\sqrt{\sum_{i=1}^{N}(k_i^{\text{out}} - \langle k \rangle)^2}} \tag{1.1.5}$$

where $\langle k \rangle$ is given in (1.1.3). This value is bounded from above by 1, meaning the total correlation such that the node with the highest in-degree also has the highest out-degree and so on, and from below by -1, where in- and out-degrees are perfectly anti-correlated. This type of degree correlation will be discussed in Chapter 2, where we find similar results to other studies in this field [VHT13, VR19, LS10, NFS+17] while we use a drastically reduced set of variables.

Degree assortativity In contrast, the second type of correlations occurs not within nodes, but between them and affects the connection probability. In a randomly connected network, the probability that any two chosen nodes are connected depends on their degree; the chance to be connected is higher when the sending node has a high out-degree and the receiving node a high in-degree. Assortativity means that this probability is altered due to some property of the nodes. It does not necessarily have to be the same property on the sending and on the receiving side. We speak of degree assortativity when those properties are degrees. If the probability of a connection between two nodes, given their degrees, is

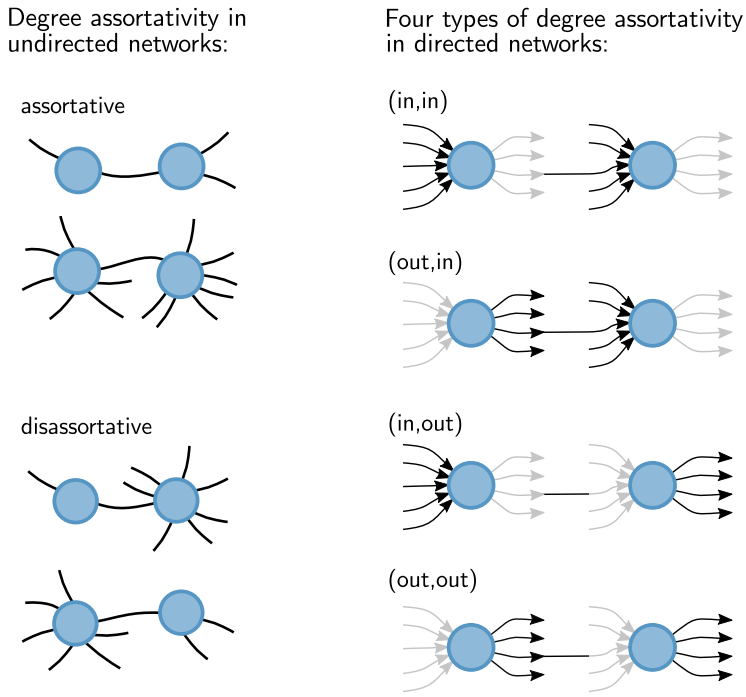

Figure 1.3: Assortativity in undirected and directed networks. An undirected network (left column) is assortative if high degree nodes are more likely to be connected to high degree nodes, and low to low, than by chance (top left). Such a network is disassortative if the opposite occurs (bottom left). In directed networks (right column) there are four possible kinds of degree assortativity. The probability of a connection is thus influenced by the number of solid black links of the sending (left) and receiving (right) node.

what one would expect by chance, the network is referred to as neutral assortative. The tendency of nodes with similar (different) degrees to prefer to mutually connect is called *positive assortative* (*negative assortative*) (Figure 1.3). Alternatively, it is common to use the terms *assortative* and *disassortative*. A directed network exhibits four different kinds of degree assortativity, each specifying whether in- or out-degree of the pre- and post-synaptic neuron are correlated. Those are (in,in)-, (in,out)-, (out,in)-, and (out,out)-assortativity (Figure 1.3). To some extent these four types can occur independently from one another. Although assortativity can be measured in different ways, we use the Pearson correlation coefficient again, which in this context may be referred to as the *assortativity coefficient* r – and for the four different types: $r(\alpha, \beta)$ with $\alpha, \beta \in [\text{in}, \text{out}]$. For its definition we form sums over edges and conveniently introduce the leading superscript s and r to differentiate between the sending and the receiving node of that edge. For example, the sending node's

in-degree of the second edge would be ${}^{s}k_2^{\text{in}}$. We then write

$$r(\alpha,\beta) = \frac{\sum_{e=1}^{N_e}({}^{s}k_e^{\alpha} - \langle {}^{s}k^{\alpha}\rangle)({}^{r}k_e^{\beta} - \langle {}^{r}k^{\beta}\rangle)}{\sqrt{\sum_{e=1}^{N_e}({}^{s}k_e^{\alpha} - \langle {}^{s}k^{\alpha}\rangle)^2}\sqrt{\sum_{e=1}^{N_e}({}^{r}k_e^{\beta} - \langle {}^{r}k^{\beta}\rangle)^2}} \tag{1.1.6}$$

where

$$\langle {}^{s}k^{\alpha}\rangle = \frac{1}{N_e}\sum_{e=1}^{N_e} {}^{s}k_e^{\alpha} \qquad \text{and} \qquad \langle {}^{r}k^{\beta}\rangle = \frac{1}{N_e}\sum_{e=1}^{N_e} {}^{r}k_e^{\beta} \tag{1.1.7}$$

Note that there are four different mean values to compute. The assortativity coefficient is bounded by 1 and -1, referring to maximal positive and negative assortativity, but typically the combination of degree distribution and number of nodes will impose narrower bounds on r.

Several authors have produced research addressing some of these cases, sometimes assuming equal in- and out-degree within each node, i.e. not isolating the two types of correlation [SKSR15, CHC+17, FFGP10, New02]. In Chapter 3 we will investigate the effects of all four of these in the absence of the other three and without correlations within nodes. Surprisingly, we will find that degree assortativity involving the out-degree of the sending neuron has no influence on the overall dynamics.

1.2 Neurons

In 1906, the Nobel Prize for Medicine was awarded to Camillo Golgi (1843-1926) and Santiago Ramón y Cajal (1852-1934) for their discovery of the neuronal network (Figure 1.4). Golgi believed that this network would basically behave like blood vessels and nodes are

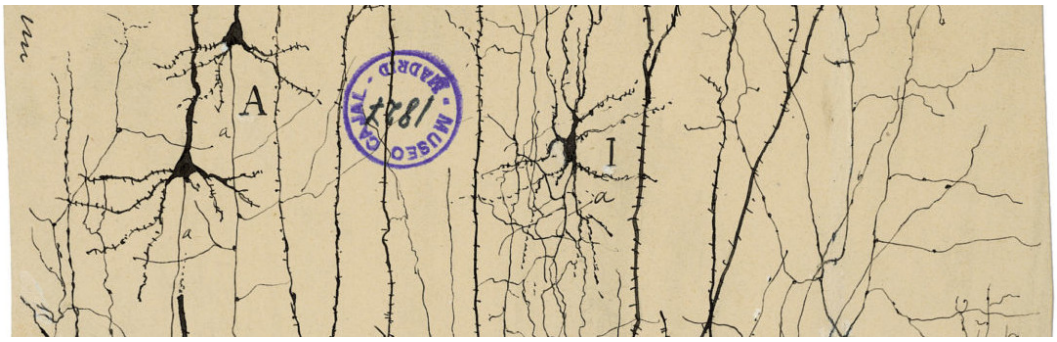

Figure 1.4: Historic drawing of nerve cells by Santiago Ramón y Cajal.[SNAD17]

just connections, whereas Ramón y Cajal correctly foresaw that those nodes, later called neurons, are separate entities with a processing function. Since their discovery, the knowl-

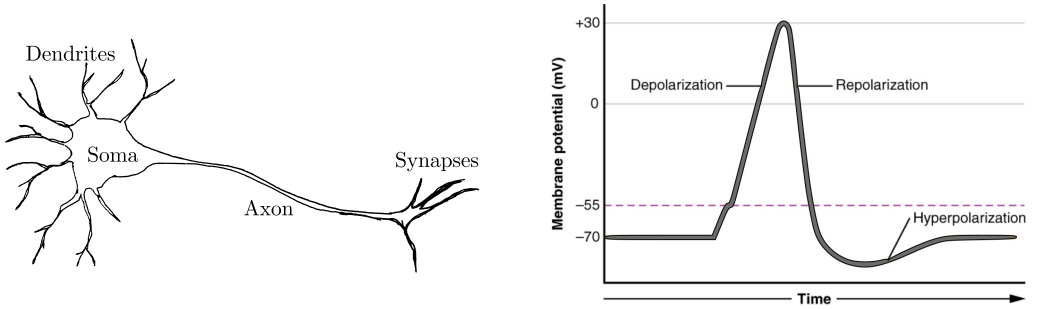

Figure 1.5: A schematic sketch of a neuron (left) and an illustration of the voltage measured between the in- and out-side of the cell membrane during a typical action potential (right). (Right image taken from wikimedia.org)

edge of neurons has become very detailed. We can now say a neuron is a living cell with the ability to process electrical signals. This is realised through ions, mainly sodium (Na^+) and potassium (K^+). Note that there are not only more than 10 distinct types of potassium channels, but also about 200 different ions involved in shaping the electrical properties of the various neurons. Signals are received through dendrites, processed in the cell body (soma) and subsequently transmitted through the axon which branches into synapses passing on the altered signal to dendrites of connected neurons (Figure 1.5 - left).

Neurons follow an all-or-none principle, meaning that their electrical output remains at resting voltage until their input exceeds a certain threshold upon which the cell reacts with an action potential (Figure 1.5 - right), which is a short electrical pulse. In this case a neuron is said to "fire". However, there are neurons which constantly fire by themselves ("tonic firing") and rather change their firing rate upon a stimulus. Stimuli can be excitatory – encouraging more action potentials, as well as inhibitory – do the opposite.

In more detail, the process of an action potential and the underlying electrochemical principles can be summarised as follows: The cell body's membrane comprises several types of ion channels with gates allowing only specific ions to pass. There is one type, which always allows K^+ to move freely in and out. In contrast, there are voltage (measured between the in- and outside of the cell) sensitive gates which open and close at certain thresholds for sodium or potassium. Inside a neuron, there are immobile, negatively charged proteins, attracting positive ions. At rest, only potassium is able to diffuse through the membrane until an electrochemical equilibrium is reached, resulting in a higher K^+ concentration inside. The cell still exhibits an overall negative charge and typically, one can measure about -70mV between the in- and outside. If a neuron is stimulated, for instance by injection of positive ions, up to a critical voltage threshold (approximately -55mV) a series of events gets triggered. Some of the gates of those ion channels are in fact voltage sensitive and one type will open for sodium (Na^+) to flood the inside following a chemical gradient.

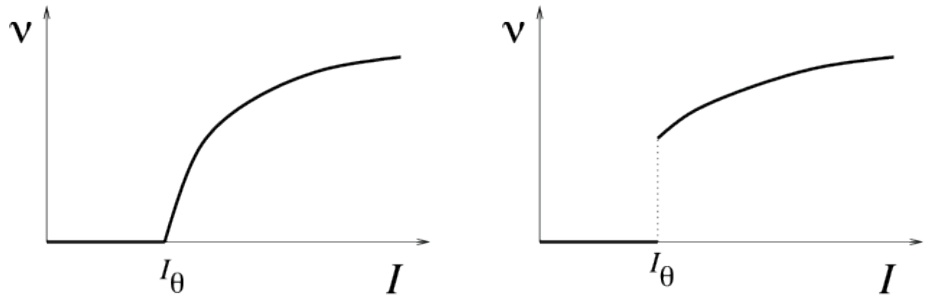

Figure 1.6: Firing frequencz ν plotted against the input current I. When stimulated with a constant synaptic current I, a neuron starts firing at a critical value I_θ. There are two types of neurons: type I can fire with an arbitrary low frequency ν (left), whereas type II exhibits a finite minimal value (right). (Image taken from neuronaldynamics.epfl.ch)

Consequently, there is a rapid voltage increase, where the cell is said to depolarise, up to about +30mV. This triggers the sodium gates to close and voltage sensitive potassium gates to open. With the outside being more negative now, K^+ ions leave the cell and in return, the membrane voltage drops, the cell repolarises, and eventually even hyperpolarises, meaning it will be more negative than at resting voltage. From here on, the K^+ gates close and a sodium-potassium pump mechanism takes over. The cell membrane contains a protein which hydrolyses ATP (adenosine triphosphate) into ADP (adenosine diphosphate) which enables the cell to move 3 Na^+ ions from the inside to the outside in exchange for 2 K^+ ions per each ATP molecule. Until the membrane voltage reaches its resting level, the neuron cannot fire again. This roughly described mechanism for the creation of an action potential can be extended and specified for the many different kinds of neurons.

Considering a constant input current, which will typically result in a neuron firing at a certain rate, there is an important classification to make. As illustrated in Figure 1.6, one type of neuron fires with an arbitrary low frequency once a current threshold is crossed (type I), whereas another type exhibits a finite minimal firing rate (type II). This distinction will be relevant in the next section when we look at neurons from a dynamical systems point of view.

1.2.1 Neuronal modelling

One of the earliest neuron models, from the beginning of the 20th century, is the integrate-and-fire model by Louis Lapicque [Abb99]:

$$I = C\frac{dV}{dt} \tag{1.2.1}$$

with V being the membrane voltage, C a cell specific capacity constant and I a stimulating current. A neuron integrates the stimulating current I until a constant threshold V_{th} is reached. T'hen it fires in a delta function like spike and the voltage is reset to zero. Lapicque derived his model from frog leg experiments without any knowledge of the underlying ion channels and as such it describes a neuron at a rather phenomenological level. However, there are major physiological issues. An integrate-and-fire neuron never settles back to its resting potential without spiking, nor does it exhibit a recovery time between action potentials. Due to its simplicity, it is still very popular today and numerous models have been derived from it, for example the quadratic integrate-and-fire model, the leaky integrate-and-fire model, the fractional-order leaky integrate-and-fire model or the exponential integrate-and-fire model[FTHVVB03].

A major extension was developed by Hodgkin and Huxley in 1952 [HH52] by taking ion gates into account. The model comprises four coupled non-linear differential equations

$$I = C\frac{dV}{dt} + \bar{g}_{\mathrm{K}} n^4 (V - V_{\mathrm{K}}) + \bar{g}_{\mathrm{Na}} m^3 h (V - V_{\mathrm{Na}}) + \bar{g}_{\mathrm{l}} (V - V_{\mathrm{l}}) \tag{1.2.2}$$

$$\frac{dn}{dt} = \alpha_n(V)(1-n) - \beta_n(V)n \tag{1.2.3}$$

$$\frac{dm}{dt} = \alpha_m(V)(1-m) - \beta_m(V)m \tag{1.2.4}$$

$$\frac{dh}{dt} = \alpha_h(V)(1-h) - \beta_h(V)h \tag{1.2.5}$$

In Eq. 1.2.2, we find three additional terms which take voltage gated potassium and sodium channels into account and a leak term modelling diffusion through the cell membrane. These terms resemble resistors in this flow of ions, hence the Hodgkin-Huxley model and all its derivatives are called conductance-based models. Here, $\bar{g}$ denotes the maximal conductance and the dimensionless variables n, m and h ranging between 0 and 1 are associated with K^+ gate activation, Na^+ gate activation and Na^+ gate inactivation, respectively. With V_{K}, V_{Na}, and V_{l} we denote the resting voltage where the respective ionic current comes to a standstill. The functions $\alpha(V)$ and $\beta(V)$ determine resting values for the gating variables and are neuron specific. This model is considered one of the greatest achievements in biophysics of the 20th century and the discoverers were awarded the Nobel Prize for Medicine in 1963. Using original parameters models a type II neuron, whereas in different parameter regimes type II behaviour can be observed.

The model remains widely used and has been generalised and extended to include various other ion channels. But we encounter a typical modelling problem here: with more incorporated realism, a model often becomes more complex; there are more parameters one has to deal with and it can be extremely difficult to study analytically. The Hodgkin-Huxley model has also been simplified by separating slow and fast variables and utilising symmetries, for instance the Morris-Lecar model [ML81] or the simple model [Izh03].

When investigating large groups of neurons, using a detailed model is computationally costly, whereas a simplification can sometimes allow analytical work and insights in processes of the real physical system. Yet, what are the most important features of a neuron we should not neglect? A possible approach is to change to rate-based models. They can be derived from conductance-based models and no longer compute the membrane voltage and action potentials, but the firing rate itself. In this thesis, we apply an intermediate model between the detailed modelling of membrane potential dynamics and the firing rate approach.

1.2.2 Canonical model

The canonical model approach of Izhikevich [Izh07] suggests we reduce the neuronal model to a minimal description, which still captures the correct dynamical transition, i.e. exhibits the same bifurcation. A bifurcation is a substantial change in dynamics, e.g. in our case transitioning from resting to periodic firing. It occurs while varying the input current, which is one parameter, thus the bifurcation is said to have codimension-1. In the following, we consider a neuron's phase space, which is the space of all its possible states. If a neuron is at rest, there has to be an attracting or stable fixed point (node) in this phase space. If it is in a state of periodically spiking, we find a limit cycle attractor. There are only four codimension-1 bifurcations that feature a transition from a stable fixed point to an attractive limit cycle (Figure 1.7). To each of the four bifurcations there is a minimal model, also called the topological normal form or canonical model. It is minimal in the sense of having the least necessary number of variables to exhibit the respective bifurcation. The authors of [HI97] state that each member of a family of models sharing the same bifurcation can be transformed by a piecewise continuous change of variables into the respective canonical model.

As mentioned earlier, there are two types of neurons with either an arbitrarily low firing rate at the transition (type I) or a positive, minimal frequency (type II). We find that only the saddle-node on an invariant circle (SNIC) bifurcation (Figure 1.7(b)) can show the behaviour of a type I neuron and the other three relate to type II neurons. In this thesis, we make the choice of studying type I neurons since its canonical model is well known and there is an elegant framework, which enables us to model a whole network of such neurons with only a small set of equations. It would be an interesting extension to this work to investigate networks of the different type II neurons.

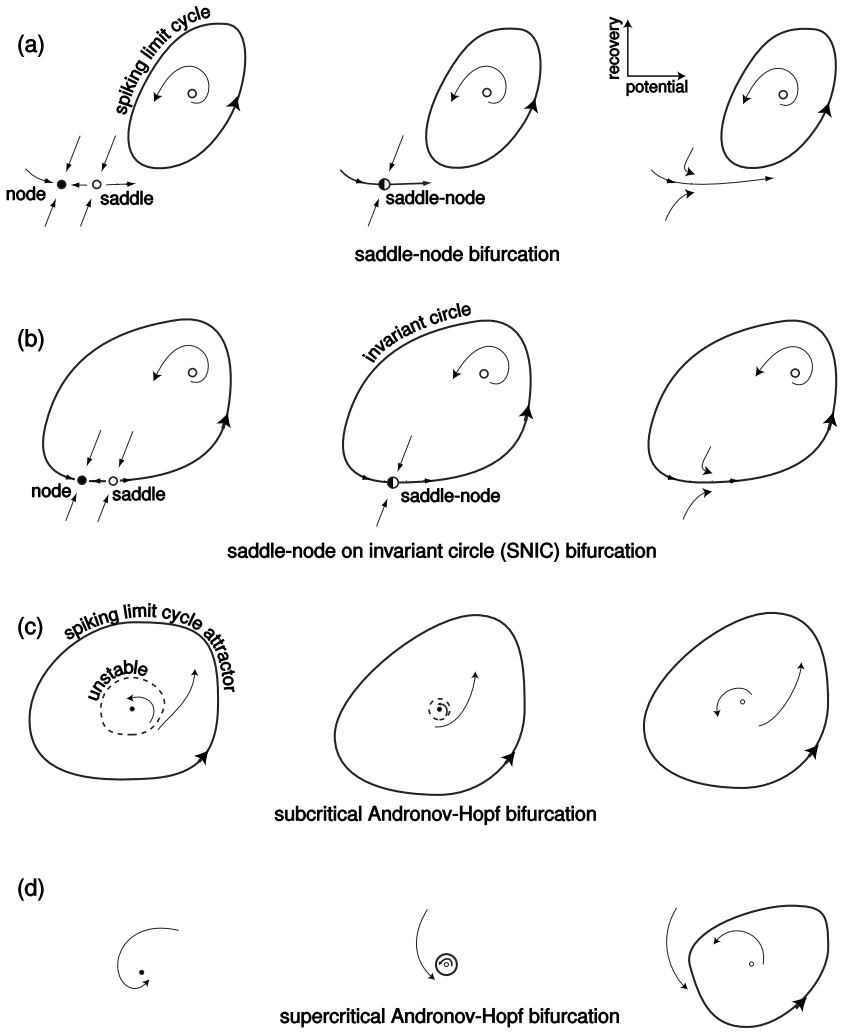

Figure 1.7: Four codimension-1 bifurcations featuring the transition from exhibiting a stable fixed point on the one side (left) over the nihilation of the very same (middle) and leaving nothing but a stable limit cycle (right). (Image taken from [Izh07])

1.2.3 Theta neuron model

We turn now to the topological normal form of a type I neuron, which is the quadratic integrate-and-fire model (QIF)[FT13]

$$\frac{dV}{dt} = V^2 + I, \qquad \text{if } V > V_{\text{peak}}, \qquad \text{then } V \to V_{\text{reset}} \tag{1.2.6}$$

with V being the membrane voltage and I the input current. The theta neuron model can be obtained from a simple variable transformation and shall be the model of choice for this

thesis, because of its continuous sinusoidal form. Thus, we will be able to use any standard ODE solver. Substituting

$$V = \tan(\theta/2) \tag{1.2.7}$$

the left hand side of Eq. (1.2.6) then reads

$$\frac{d\tan\left(\frac{\theta}{2}\right)}{d\theta} = \frac{1}{2\cos^2\left(\frac{\theta}{2}\right)}\frac{d\theta}{dt} \tag{1.2.8}$$

Thus, we write for the dynamical equation of θ

$$\frac{d\theta}{dt} = 2\left(\sin^2\left(\frac{\theta}{2}\right) + \cos^2\left(\frac{\theta}{2}\right) I\right) \tag{1.2.9}$$

and by using trigonometric identities we arrive at the so-called Theta Neuron model or Ermentrout-Kopell canonical model [EK86]

$$\frac{d\theta}{dt} = (1-\cos\theta) + (1+\cos\theta)I \tag{1.2.10}$$

The variable θ can be understood as a state variable and per definition a neuron described by this model is said to fire at $\theta = \pi$. Using Eq. (1.2.7), we can relate this to the QIF model: $V_{\text{peak}} = +\infty$ and $V_{\text{reset}} = -\infty$. The parameter I still models the input current. Figure 1.8 illustrates how varying the current passing 0 changes the dynamical behaviour. For negative currents, Eq. (1.2.10) has two roots, i.e. two stationary points. One is stable (node) and the other unstable (saddle). In this scenario, the neuron approaches the stable fixed point and rests. It requires a positive, sufficiently strong pulse to push it beyond the unstable saddle point, from where θ will further increase through π causing an action potential and eventually return to the node. With a constant positive current, the system has no fixed points. The instantaneous velocity $d\theta/dt$ is strictly positive and the constantly increasing θ will periodically reach the firing value π. Approaching the transition with a positive current ($I > 0; I \to 0$), the minimal velocity is still positive, but becomes arbitrary small. In this regime, a theta neuron slows down around $\theta = 0$, thereby increasing the time between its spikes. Hence its firing rate gets smaller and smaller. This perfectly resembles the behaviour of a type I neuron.

1.2.4 A network of excitable theta neurons

We consider the input current to be the sum of two terms: an internal stimulus or intrinsic excitability η and an incoming current I. With the excitability parameter, we can model whether a neuron is firing on its own, needs only little external stimulus or is hardly excitable

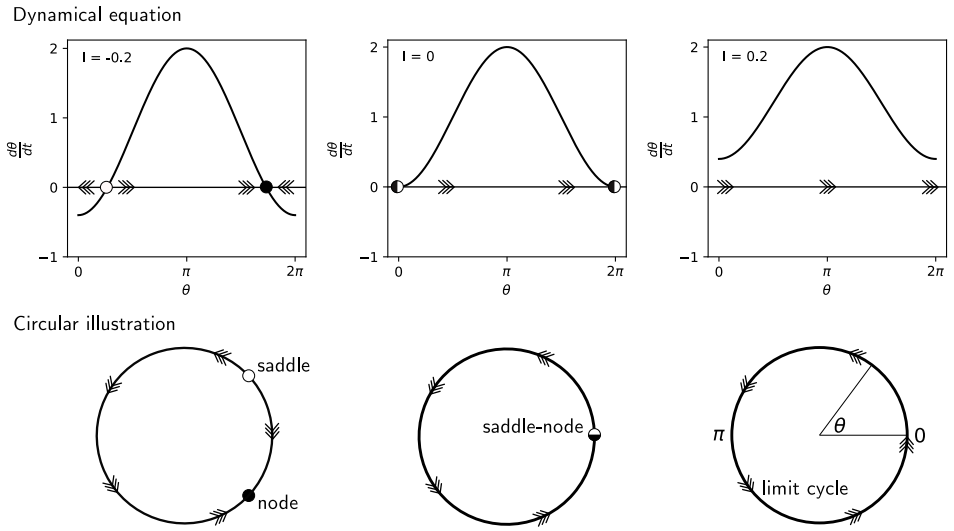

Figure 1.8: Saddle-node on an invariant circle (SNIC) bifurcation of the theta neuron model. From left to right, the bifurcation parameter I is varied passing 0 and thus changing the dynamics substantially. The upper plots show the dynamical equation for the respective input current whereas the figures below illustrate the 1-dimensional phase plane in polar coordinates with an arbitrary radius.

at all. A network of N theta neurons can be written as

$$\frac{d\theta_i}{dt} = (1 - \cos\theta_i) + (1 + \cos\theta_i)\,(\eta_i + I_i) \tag{1.2.11}$$

with $i = 1, 2, \ldots, N$. Note that this is a heterogeneous model and each neuron has its individual excitability parameter η_i and incoming current I_i. For our studies of self-sustained patterns in recurrent neuronal networks, the external stimulus is the sum of output currents of connected neurons:

$$I_i = \frac{K}{\langle k \rangle} \sum_{j=1}^{N} A_{ij} P_n(\theta_j) \tag{1.2.12}$$

An entry A_{ij} of the adjacency matrix A is either 1 if neuron j's output is connected to neuron i or 0 otherwise, as described in Section 1.1. Typically, neuron j's synaptic pulse $P_n(\theta_j)$ is modelled by

$$P_n(\theta) = a_n(1 - \cos\theta)^n \qquad \text{with } a_n \text{ such that} \qquad \int_0^{2\pi} P_n(\theta)d\theta = 1 \tag{1.2.13}$$

where the parameter n determines the sharpness of the pulse (Figure 1.9). For better control over the external stimulus, we normalise the sum in Eq.(1.2.12) with the average number

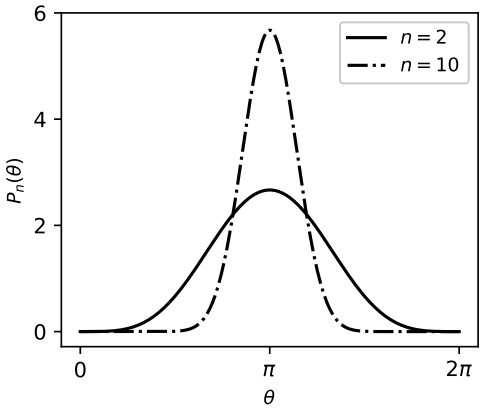

Figure 1.9: The pulse function $P_n(\theta)$ used in the theta neuron model plotted for different values of the sharpness parameter n.

of connections, which is the mean degree $\langle k \rangle$, and multiply the result by a homogeneous coupling strength K. This value could as well be modelled as being heterogeneous, for example to model a spatially extended network.

With our chosen pulse function and a particular distribution of intrinsic excitability values, the system Eq. (1.2.11) is known to be amenable to the use of the Ott/Antonsen theory [OA08, OA09, LBS13]. It has its origins in the theory of coupled oscillators and was first applied to the Kuramoto model. When considering several such systems, a probability density function for the state of each neuron can be introduced. The theory states that every density function will approach an invariant manifold described by the Ott/Antonsen ansatz. As we use two slightly different mean field theories in Chapter 2 and Chapter 3, their derivation will be given as we use them.

1.2.5 Order parameter and firing frequency

Looking at large networks, we do not analyse individual neurons, but compute meaningful observables of the network dynamics. The first to mention is the complex Kuramoto order parameter z

$$z = \frac{1}{N} \sum_{j=1}^{N} z_j \qquad \text{with} \qquad z_j = e^{i\theta_j} \tag{1.2.14}$$

where i denotes the imaginary unit. Its absolute value $|z|$ being in the range $[0, 1]$ correlates with the synchronicity of the network. The larger $|z|$ the more synchronized the network is. In a rather synchronised scenario the argument of z indicates the state of the majority of

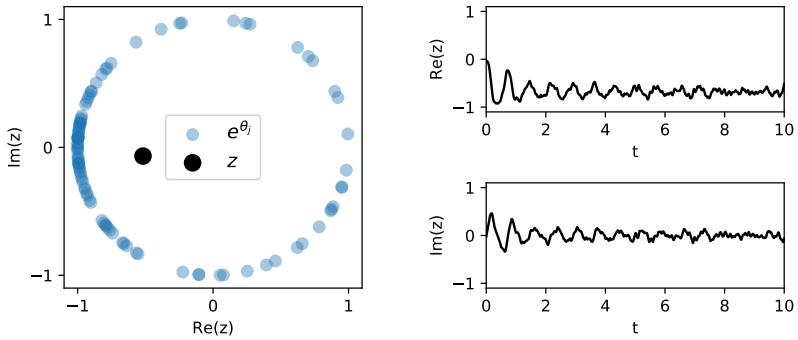

Figure 1.10: The left panel shows 100 theta neurons after 10 time units mapped into the complex plane (blue circles) and their mean value – the Kuramoto order parameter (black circle). The time evolution of z is illustrated on the right side (top: real part, bottom: imaginary part). Although the underlying neuronal activity is ongoing, for large systems the order parameter will often be steady or move in periodic orbits. (Simulation parameters: $N = 1000$; $k^{\text{in}}, k^{\text{out}} \in [300, 600]$; $P(k) \sim k^{-3}$; $K = 1.6$; $\eta_i \approx -2\ \forall i$; $n = 4$)

neurons (Figure 1.10). This relates to the state θ of an individual theta neuron, i.e. firing at π and so on. For a more in-depth analysis of the order parameter in the context of Kuramoto oscillators see [Pet19].

The authors of [MPR15] derive a mapping between the Kuramoto order parameter and the mean firing rate and membrane potential. Their map is only exact in a system to which the Ott/Antonsen theory is applicable – such as the systems we will investigate. Given z we calculate the variable w

$$w = \frac{1 - \bar{z}}{1 + \bar{z}} \tag{1.2.15}$$

where $\bar{z}$ denotes the complex conjugate of z, from which we gain

$$f = \frac{1}{\pi}\,\mathrm{Re}(w) \qquad \text{and} \qquad V = \mathrm{Im}(w) \tag{1.2.16}$$

with f being the mean firing frequency and V the mean membrane potential of the original QIF neurons. In Chapter 2 and Chapter 3 we will use the mean firing frequency as our primary observable of the system.

We have seen to what extent a theta neuron is a valid representation of a real neuron and discussed the two types of degree correlation we will study in the following chapters. How to generate networks with the desired correlation in them will be covered in the respective section. We now turn to the study of correlations between in- and out-degree within each neuron.

Chapter 2

Within-neuron degree correlations

In this chapter, we consider the effects of correlations between the in- and out-degrees of individual neurons on the dynamics of a network of neurons. Using theta neurons, we derive a set of coupled differential equations for the expected dynamics of neurons with the same in-degree. A Gaussian copula is used to introduce correlations between a neuron's in- and out-degree and with numerical bifurcation analysis we determine the effects of these correlations on the network's dynamics. We also investigate the propensity of various two- and three-neuron motifs to occur as correlations are varied and give a plausible explanation for the observed changes in dynamics.

The content of this section is an altered version of the publication [LB20].

2.1 Introduction

Determining the effects of a network's structure on its dynamics is an issue of great interest, particularly in the case of a network of neurons [Rox11, SKSR15, NFS+17, MHT17]. In Chapter 1 we introduced degrees and their distribution. Since neurons form *directed* synaptic connections, a neuron has both an in-degree — the number of neurons connected to it, and an out-degree — the number of neurons it connects to. In this chapter, we present a framework for investigating the effects of correlations, both positive and negative, between these two quantities. To isolate the effects of correlations we assume no other structure in the networks, that is random connectivity based on the neurons' degrees.

A number of other authors have considered this issue and we now summarise relevant aspects of their results. LaMar and Smith [LS10] investigated directed networks of identical pulse-coupled phase oscillators and mostly concentrated on the probability that the

network fully synchronises, and the time taken to do so. Vasquez et al. [VHT13] studied binary neurons whose states were updated at discrete times, and found that negative degree correlations stabilised a low firing rate state for excitatory coupling. A later paper [MHT17] dealt with more realistic spiking neurons, had a mix of excitatory and inhibitory neurons, and concentrated more on the network's response to transient stimuli, as well as on the analysis of network properties such as the mean shortest path. Several authors have considered networks for which the in- and out-degrees of a neuron are equal, thereby inducing positive correlations between them [SKSR15, KSR17].

Vegué and Roxin [VR19] simulated large networks of both excitatory and inhibitory leaky integrate-and-fire neurons and used a mean-field formalism to determine steady state distributions of firing rates within neural populations. They studied the effects of within-neuron degree correlations for the excitatory to excitatory connections, and varied the probability of inhibitory to excitatory connections in order to create a "balanced state". Nykamp et al. [NFS$^+$17] also considered large networks of both excitatory and inhibitory neurons and used a Wilson-Cowan type firing rate model to investigate the effects of within-neuron degree correlations. They showed that once correlations were included, the dynamics were effectively four-dimensional, in contrast to the two-dimensional dynamics expected from a standard rate-based excitatory/inhibitory network. They also related the degree distributions to cortical motifs. Experimental evidence for within-neuron degree correlations was given in [VPR17].

This chapter is structured as follows: in Sec. 2.2, we present the model network and summarise the analysis of [CHC$^+$17], showing that under certain assumptions the network dynamics can be described by a coupled set of ordinary differential equations, one for each in-degree. In Sec. 2.3, we discuss how to generate correlated in- and out-degrees using a Gaussian copula. Our model involves sums over all distinct in-degrees, and in Sec. 2.4, we present a computationally efficient method for evaluating these sums, in analogy with Gaussian quadrature. Our main results are described in Sec. 2.5 and we show in Sec. 2.6 that they also occur in networks of more realistic Morris-Lecar spiking neurons. An analysis of how motifs change under the influence of degree correlations is given in Sec. 2.7. We conclude this chapter in Sec. 2.8.

2.2 Theta neuron mean field using a degree dependent Ott/Antonsen ansatz

We consider the same model of pulse-coupled theta neurons as in [CHC$^+$17].

The governing equations are

$$\frac{d\theta_i}{dt} = 1 - \cos\theta_i + (1 + \cos\theta_i)(\eta_i + I_i) \tag{2.2.1}$$

for $i = 1, 2 \dots N$, where the phase angle θ_i characterises the state of neuron i, which fires an action potential as θ_i increases through π, the synaptic current I_i reads

$$I_i = \frac{K}{\langle k \rangle} \sum_{j=1}^{N} A_{ij} P_n(\theta_j) \tag{2.2.2}$$

K is the strength of connections within the network, $A_{ij} = 1$ if there is a connection from neuron j to neuron i and $A_{ij} = 0$ otherwise, $\langle k \rangle$ is the average degree, $\sum_{i,j} A_{ij}/N$, and $P_n(\theta) = a_n(1 - \cos\theta)^n$ where a_n is chosen such that $\int_0^{2\pi} P_n(\theta) d\theta = 1$. The function $P_n(\theta_j)$ models the pulse of current emitted by neuron j when it fires and can be made arbitrarily "spike-like" and localised around $\theta_j = \pi$ by increasing n.

The parameter η_i is the input current to neuron i in the absence of coupling and the η_i are independently and randomly chosen from a Lorentzian distribution

$$g(\eta) = \frac{\Delta/\pi}{(\eta - \eta_0)^2 + \Delta^2} \tag{2.2.3}$$

Chandra et al. [CHC+17] considered the limit of large N and assumed that the network can be characterised by two functions: firstly, a degree distribution $P(\boldsymbol{k})$, normalised so that $\sum_{\boldsymbol{k}} P(\boldsymbol{k}) = N$, where $\boldsymbol{k} = (k_{in}, k_{out})$ and k_{in} and k_{out} are the in- and out-degrees, respectively of a neuron with degree $\boldsymbol{k}$. Secondly, an assortativity function $a(\boldsymbol{k}' \to \boldsymbol{k})$ giving the probability of a connection from a neuron with degree $\boldsymbol{k}'$ to one with degree $\boldsymbol{k}$. While [CHC+17] investigated the effects of varying $a(\boldsymbol{k}' \to \boldsymbol{k})$, here we consider the default value for this assortativity function (i.e. its value expected by chance, see (2.2.11)) and investigate the effects of varying correlations between k_{in} and k_{out} as specified by the degree distribution $P(\boldsymbol{k})$. We emphasise that we are only considering *within-neuron* degree correlations in this chapter and are not considering degree assortativity, which refers to the probability of neurons with specified degrees being connected to one another [CHC+17, RO14] as in Chapter 3.

In the limit $N \to \infty$, the network can be described by a probability distribution $f(\theta, \eta | \boldsymbol{k}, t)$, where $f(\theta, \eta | \boldsymbol{k}, t) d\theta \; d\eta$ is the probability that the phase angle of a neuron with degree $\boldsymbol{k}$ is in $[\theta, \theta + d\theta]$ and its intrinsic excitability η in $[\eta, \eta + d\eta]$ at time t. This distribution satisfies the continuity equation

$$\frac{\partial f}{\partial t} + \frac{\partial}{\partial \theta}(v f) = 0 \tag{2.2.4}$$

where v is the continuum version of the right hand side of (2.2.1):

$$\begin{aligned} v(\theta, \boldsymbol{k}, \eta, t) = {} & 1 - \cos\theta + (1 + \cos\theta) \\ & \times \left[\eta + \frac{K}{\langle k \rangle} \sum_{\boldsymbol{k}'} P(\boldsymbol{k}') a(\boldsymbol{k}' \to \boldsymbol{k}) \int_{-\infty}^{\infty} \int_0^{2\pi} f(\theta', \eta' | \boldsymbol{k}', t) P_n(\theta') d\theta' \; d\eta' \right] \end{aligned} \tag{2.2.5}$$

The system (2.2.4)-(2.2.5) is amenable to the use of the Ott/Antonsen ansatz [OA08, OA09] and using standard techniques [LBS13, Lai14a, Lai16, CB19] one can show that the long-time dynamics of the system is described by

$$\begin{aligned}\frac{\partial b(\boldsymbol{k},t)}{\partial t} &= \frac{-i(b(\boldsymbol{k},t)-1)^2}{2} + \frac{(b(\boldsymbol{k},t)+1)^2}{2} \\ &\quad \times \left[-\Delta + i\eta_0 + \frac{iK}{\langle k\rangle}\sum_{\boldsymbol{k}'} P(\boldsymbol{k}')a(\boldsymbol{k}'\to\boldsymbol{k})G(\boldsymbol{k}',t)\right]\end{aligned} \tag{2.2.6}$$

where (having chosen $n = 2$)

$$G(\boldsymbol{k}',t) = 1 - \frac{2(b(\boldsymbol{k}',t)+\bar{b}(\boldsymbol{k}',t))}{3} + \frac{b(\boldsymbol{k}',t)^2+\bar{b}(\boldsymbol{k}',t)^2}{6} \tag{2.2.7}$$

The quantity

$$b(\boldsymbol{k},t) = \int_{-\infty}^{\infty}\int_0^{2\pi} f(\theta,\eta|\boldsymbol{k},t)e^{i\theta}d\theta\ d\eta \tag{2.2.8}$$

can be regarded as a complex-valued "order parameter" for neurons with degree $\boldsymbol{k}$ at time t. The function $G(\boldsymbol{k}',t)$ can be regarded as the output current from neurons with degree $\boldsymbol{k}'$, and its form results from rewriting the pulse function $P_n(\theta)$ in terms of $b(\boldsymbol{k}',t)$. [For general n, $G(\boldsymbol{k}',t)$ is the sum of a degree-n polynomial in $b(\boldsymbol{k}',t)$ and in $\bar{b}(\boldsymbol{k}',t)$ (the conjugate of $b(\boldsymbol{k}',t)$) [Lai14a, LBS13]. One can take the limit $n\to\infty$, corresponding to a delta-function like pulse, and obtain $G(\boldsymbol{k}',t) = (1-|b(\boldsymbol{k}',t)|^2)/(1+b(\boldsymbol{k}',t)+\bar{b}(\boldsymbol{k}',t)+|b(\boldsymbol{k}',t)|^2)$.] Note that the parameters of the Lorenztian (2.2.3) appear in (2.2.6) as a result of evaluating the integral over η' in (2.2.5). The equation (2.2.6) describes only the long-time asymptotic behaviour of the network (2.2.1), on the "Ott/Antonsen manifold", and thus may not fully describe transients from arbitrary initial conditions, nor the effects of stimuli which move the network off this manifold.

One can also marginalise $f(\theta,\eta|\boldsymbol{k},t)$ over η to obtain the distribution of θ for each $\boldsymbol{k}$ and t:

$$p_\theta(\theta|\boldsymbol{k},t) = \frac{1-|b(\boldsymbol{k},t)|^2}{2\pi\{1-2|b(\boldsymbol{k},t)|\cos\left[\theta-\arg(b(\boldsymbol{k},t))\right]+|b(\boldsymbol{k},t)|^2\}} \tag{2.2.9}$$

a unimodal function with maximum at $\theta = \arg(b(\boldsymbol{k},t))$. The firing rate of neurons with degree $\boldsymbol{k}$ is equal to the flux through $\theta = \pi$:

$$\begin{aligned}f(\boldsymbol{k},t) &= 2p_\theta(\pi|\boldsymbol{k},t) \\ &= \frac{1-|b(\boldsymbol{k},t)|^2}{\pi\{1+2|b(\boldsymbol{k},t)|\cos\left[\arg(b(\boldsymbol{k},t))\right]+|b(\boldsymbol{k},t)|^2\}} \\ &= \frac{1}{\pi}\mathrm{Re}\left(\frac{1-\bar{b}(\boldsymbol{k},t)}{1+\bar{b}(\boldsymbol{k},t)}\right)\end{aligned} \tag{2.2.10}$$

where we have used the fact that $d\theta/dt = 2$ when $\theta = \pi$.

Suppose our network has neutral assortativity, that is the probability for two neurons to be connected scales linearly with the out-degree of the sending one and the in-degree of the receiving one. Then [RO14, CHC+17]

$$a(\boldsymbol{k}' \to \boldsymbol{k}) = \frac{k'_{out} k_{in}}{N\langle k \rangle} \tag{2.2.11}$$

and

$$\begin{aligned} &\sum_{k'_{in}} \sum_{k'_{out}} P(k'_{in}, k'_{out}, \hat{\rho}) a(\boldsymbol{k}' \to \boldsymbol{k}) G(k'_{in}, k'_{out}, t) \\ &= \frac{k_{in}}{N\langle k \rangle} \sum_{k'_{in}} \sum_{k'_{out}} P(k'_{in}, k'_{out}, \hat{\rho}) k'_{out} G(k'_{in}, k'_{out}, t) \end{aligned} \tag{2.2.12}$$

where we write $P(k'_{in}, k'_{out}, \hat{\rho})$ instead of $P(\boldsymbol{k}')$ from now on, where $\hat{\rho}$ is a parameter used to calibrate the desired correlation between k'_{in} and k'_{out}, defined below in (2.3.2). This quantity is proportional to the input to a neuron with degree (k_{in}, k_{out}) from other neurons within the network but it is clearly independent of k_{out}, hence the state of a neuron with degree (k_{in}, k_{out}) must also be independent of k_{out}, and thus G must be independent of k'_{out}. Therefore the expression in (2.2.12) can be written

$$\frac{k_{in}}{N\langle k \rangle} \sum_{k'_{in}} Q(k'_{in}, \hat{\rho}) G(k'_{in}, t) \tag{2.2.13}$$

where

$$Q(k'_{in}, \hat{\rho}) \equiv \sum_{k'_{out}} P(k'_{in}, k'_{out}, \hat{\rho}) k'_{out} \tag{2.2.14}$$

The function Q can be thought of as a k'_{in}-dependent mean of k'_{out} which also depends on the correlations between k'_{in} and k'_{out}.

Our model equations are thus

$$\begin{aligned} \frac{\partial b(k_{in}, t)}{\partial t} &= \frac{-i(b(k_{in}, t) - 1)^2}{2} + \frac{(b(k_{in}, t) + 1)^2}{2} \\ &\times \left[-\Delta + i\eta_0 + \frac{iK k_{in}}{N\langle k \rangle^2} \sum_{k'_{in}} Q(k'_{in}, \hat{\rho}) G(k'_{in}, t) \right] \end{aligned} \tag{2.2.15}$$

where k_{in} takes on integer values between the minimum and maximum in-degrees. The correlation between in- and out-degrees of a neuron is controlled by $\hat{\rho}$ (see Sec.2.3). It appears as a parameter in (2.2.14).

It is interesting to compare (2.2.14)-(2.2.15) with the heuristic rate equation in [NFS+17].

These authors characterised a neuron by its "f-I curve" — a nonlinear function transforming an input current into a firing rate. They concluded that the input current to a neuron is proportional to two quantities: (i) its in-degree, and (ii) the sum over in- and out-degrees of presynaptic neurons of the product of the joint degree distribution, the out-degree of the presynaptic neuron, and the "output" of presynaptic neurons. We also find this form of equation.

The transformation $V = \tan(\theta/2)$ maps a theta neuron to a quadratic integrate-and-fire (QIF) neuron with threshold and resets of $\pm\infty$. For the special case delta-function like sharp pulses with $n = \infty$ in $P(\theta)_n$ one can derive an equivalent pair of real equations rather than the single equation (2.2.15) where the two real variables are the mean voltage and firing rate of the QIF neurons with a specific in-degree [MPR15].

2.3 Generating correlated in- and out-degrees

We now turn to the problem of deriving $P(k'_{in}, k'_{out}, \hat{\rho})$ and thus $Q(k'_{in}, \hat{\rho})$. For simplicity, we choose the distributions of both in- and out-degrees to be the same, namely power law distributions with exponent -3, truncated below and above at degrees a and b, respectively. (Evidence for power law distributions in the human brain is given in [ECC+05], for example.) The probability distribution function of either in- or out-degree k is

$$p(k) = \begin{cases} \left(\frac{2a^2b^2}{b^2-a^2}\right) k^{-3} & a \leqslant k \leqslant b \\ 0 & \text{otherwise} \end{cases} \tag{2.3.1}$$

where the normalisation factor results from approximating the sum from a to b by an integral. (The approximation improves as a and b are both increased.) We want to introduce correlations between the in- and out-degree of a neuron, while retaining these marginal distributions. We do this using a Gaussian copula [Nel07]. The joint probability function $P(k'_{in}, k'_{out}, \hat{\rho})$ can be computed numerically, but below we give an analytical expression for the chosen marginal functions $p(k_{in})$ and $p(k_{out})$. Using the cumulative density function of the degree probability function we can relate degrees to the cumulative density function of a correlated bivariate normal distribution. The mixed partial derivative of this remapped cumulative density function is the desired probability function $P(k'_{in}, k'_{out}, \hat{\rho})$ (see Fig.2.1).

The correlated bivariate normal distribution with zero mean is

$$\begin{aligned} f(x, y, \hat{\rho}) &= \frac{1}{2\pi\sqrt{|\Sigma|}} e^{-(\mathbf{x}^T \Sigma^{-1} \mathbf{x})/2} \\ &= \frac{1}{2\pi\sqrt{1-\hat{\rho}^2}} e^{-(x^2 - 2\hat{\rho}xy + y^2)/[2(1-\hat{\rho}^2)]} \end{aligned} \tag{2.3.2}$$

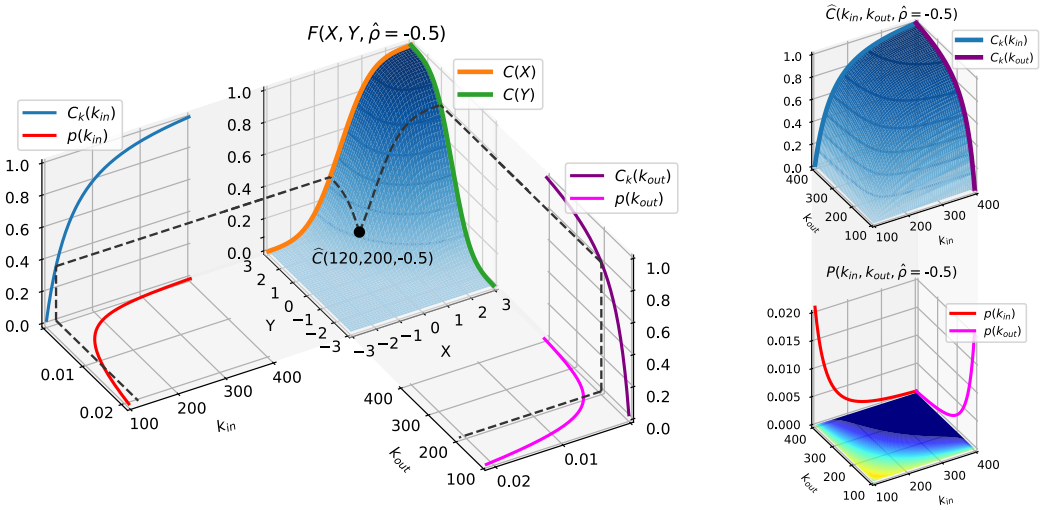

Figure 2.1: A schematic illustration of the construction of a correlated probability function $P(k_{in}, k_{out}, \hat{\rho} = -0.5)$ using a Gaussian copula. On the left panel, the degree tuple $(120, 200)$ is mapped via C_k and C^{-1} onto $F(X, Y, -0.5)$. Setting $F(X, Y, -0.5) = \widehat{C}(k_{in}, k_{out}, -0.5)$ gives a remapped function in degree space shown on the right top. Below, the colormap shows the logarithm of $P(k_{in}, k_{out}, -0.5)$, which is the mixed partial derivative of $\widehat{C}(k_{in}, k_{out}, -0.5)$. The same plot can be found on the left hand side of Figure 2.2. The summation over either degree axis yields the original marginal degree probability $p(k)$.

where

$$\mathbf{x} \equiv \begin{pmatrix} x \\ y \end{pmatrix} \qquad \Sigma = \begin{pmatrix} 1 & \hat{\rho} \\ \hat{\rho} & 1 \end{pmatrix} \tag{2.3.3}$$

and the scalar $\hat{\rho} \in (-1, 1)$ is the correlation between x and y. The variables x and y have no physical meaning. We use the copula as a way of deriving an analytic expression for $P(k'_{in}, k'_{out}, \hat{\rho})$ for which the correlations between k'_{in} and k'_{out} can be varied systematically.

The marginal distributions and therefore the cumulative distribution functions for x and y are the same:

$$\tilde{p}(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2} \qquad \text{and} \qquad C(x) = [1 + \text{erf}(x/\sqrt{2})]/2 \tag{2.3.4}$$

We define the cumulative distribution function of f:

$$F(X, Y, \hat{\rho}) = \int_{-\infty}^{Y} \int_{-\infty}^{X} f(x, y, \hat{\rho}) dx \; dy \tag{2.3.5}$$

Further, we have the cumulative distribution function for a degree k:

$$C_k(k) = \int_a^k \left(\frac{2a^2b^2}{b^2 - a^2} \right) s^{-3} ds = \frac{b^2(k^2 - a^2)}{k^2(b^2 - a^2)} \tag{2.3.6}$$

where we have treated k as a continuous variable and again approximated a sum by an integral. We thus have the joint cumulative distribution function for k_{in} and k_{out}

$$\begin{aligned} \widehat{C}(k_{in}, k_{out}, \hat{\rho}) &= F(C^{-1}(C_k(k_{in})), C^{-1}(C_k(k_{out})), \hat{\rho}) \\ &= \int_{-\infty}^{C^{-1}(C_k(k_{out}))} \int_{-\infty}^{C^{-1}(C_k(k_{in}))} f(x, y, \hat{\rho}) dx \; dy \end{aligned} \tag{2.3.7}$$

This is schematically illustrated on the left hand side of Figure 2.1.

The joint degree distribution for k_{in} and k_{out} is then

$$\begin{aligned} P(k_{in}, k_{out}, \hat{\rho}) =& \frac{\partial^2}{\partial k_{in} \partial k_{out}} \widehat{C}(k_{in}, k_{out}, \hat{\rho}) \\ =& \{C^{-1}[C_k(k_{in})]\}' \{C^{-1}[C_k(k_{out})]\}' \\ & \times f\{C^{-1}[C_k(k_{in})], C^{-1}[C_k(k_{out}])\}, \hat{\rho}\} \end{aligned} \tag{2.3.8}$$

where the primes indicate differentiation with respect to the relevant k (Right hand side of Figure 2.1). Now,

$$C^{-1}(x) = \sqrt{2} \; \text{erf}^{-1}(2x - 1) \tag{2.3.9}$$

so

$$C^{-1}[C_k(k)] = \sqrt{2} \; \text{erf}^{-1} \left(\frac{2b^2(k^2 - a^2)}{k^2(b^2 - a^2)} - 1 \right) \tag{2.3.10}$$

and

$$\{C^{-1}[C_k(k)]\}' = \sqrt{\frac{\pi}{2}} \exp \left[\left\{ \text{erf}^{-1} \left(\frac{2b^2(k^2 - a^2)}{k^2(b^2 - a^2)} - 1 \right) \right\}^2 \right] \frac{4a^2b^2}{(b^2 - a^2)k^3} \tag{2.3.11}$$

Substituting these into (2.3.8)

$$\begin{aligned} P(k_{in}, k_{out}, \hat{\rho}) =& \frac{4a^4b^4}{\sqrt{1 - \hat{\rho}^2}(b^2 - a^2)^2 k_{in}^3 k_{out}^3} \\ & \times \exp \left\{ \frac{\hat{\rho} C^{-1}[C_k(k_{in})] C^{-1}[C_k(k_{out})]}{1 - \hat{\rho}^2} \right\} \\ & \times \exp \left[\frac{-\hat{\rho}^2 \left(\{C^{-1}[C_k(k_{in})]\}^2 + \{C^{-1}[C_k(k_{out})]\}^2 \right)}{2(1 - \hat{\rho}^2)} \right] \end{aligned} \tag{2.3.12}$$

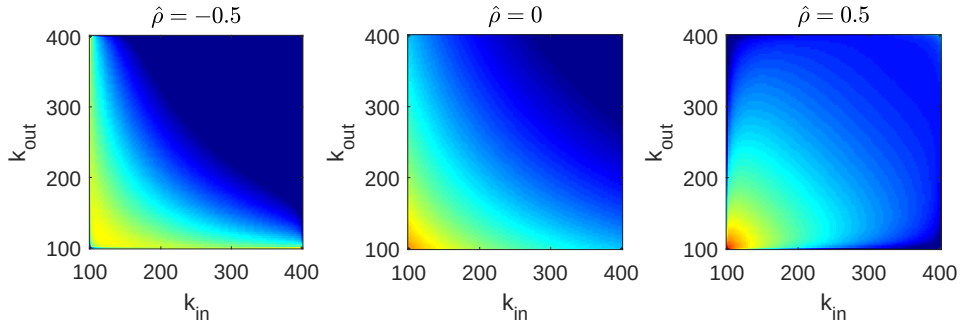

Figure 2.2: The logarithm of $P(k_{in}, k_{out}, \hat{\rho})$ is shown for three different values of $\hat{\rho}$ (red: larger P, blue: smaller P). $a = 100, b = 400$.

and simplifying we find

$$P(k_{in}, k_{out}, \hat{\rho}) = \frac{p(k_{in})p(k_{out})}{\sqrt{1-\hat{\rho}^2}} \exp\left\{\frac{\hat{\rho}C^{-1}[C_k(k_{in})]C^{-1}[C_k(k_{out})]}{1-\hat{\rho}^2}\right\} \times \exp\left[\frac{-\hat{\rho}^2\left(\{C^{-1}[C_k(k_{in})]\}^2 + \left\{C^{-1}[C_k(k_{out})]\right\}^2\right)}{2(1-\hat{\rho}^2)}\right] \tag{2.3.13}$$

Note that for $\hat{\rho} = 0$, this simplifies to $p(k_{in})p(k_{out})$, as expected. Examples of $P(k_{in}, k_{out}, \hat{\rho})$ for different $\hat{\rho}$ are shown in Fig. 2.2. Both Zhao et al. [ZBNN11] and LaMar and Smith [LS10] used Gaussian copulas to create networks with correlated in- and out-degrees as done here, but did not derive an analytical expression of the form (2.3.13).

We need to relate $\hat{\rho}$, a parameter in (2.3.13), to ρ, the Pearson's correlation coefficient between in- and out-degrees of a neuron (note: not between two connected neurons). We have

$$\rho = \frac{\tilde{\Sigma}P(k_{in}, k_{out}, \hat{\rho})(k_{in} - \langle k\rangle)(k_{out} - \langle k\rangle)}{\sqrt{\tilde{\Sigma}P(k_{in}, k_{out}, \hat{\rho})(k_{in} - \langle k\rangle)^2}\sqrt{\tilde{\Sigma}P(k_{in}, k_{out}, \hat{\rho})(k_{out} - \langle k\rangle)^2}} \tag{2.3.14}$$

where $\tilde{\Sigma}$ indicates a sum over all k_{in} and k_{out}. ρ as a function of $\hat{\rho}$ is shown in Fig. 2.3. We see that the relationship is monotonic, and while it is possible to obtain values of ρ close to 1, the lower limit is approximately -0.6. By varying $\hat{\rho}$ in (2.2.15), we can thus investigate the effects of varying the correlation coefficient ρ between in- and out-degrees of a neuron on the dynamics of a network. Note that for the distributions used here, we treat k as a continuous variable with mean value $\langle k\rangle = 2ab/(b+a)$.

Keeping in mind the normalisation $\sum_{\boldsymbol{k}} P(\boldsymbol{k}) = N$, we write $Q(k'_{in}, \hat{\rho})$ as

$$Q(k'_{in}, \hat{\rho}) = N \sum_{k'_{out}=a}^{b} P(k'_{in}, k'_{out}, \hat{\rho})k'_{out} \tag{2.3.15}$$

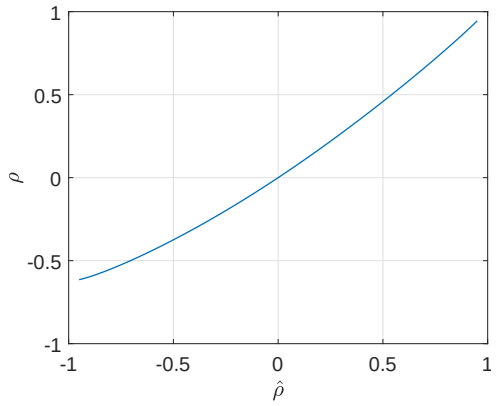

Figure 2.3: Correlation coefficient between in- and out-degrees, ρ, as a function of the correlation coefficient in the Gaussian copula, $\hat{\rho}$. Parameters: $a = 100, b = 400$.

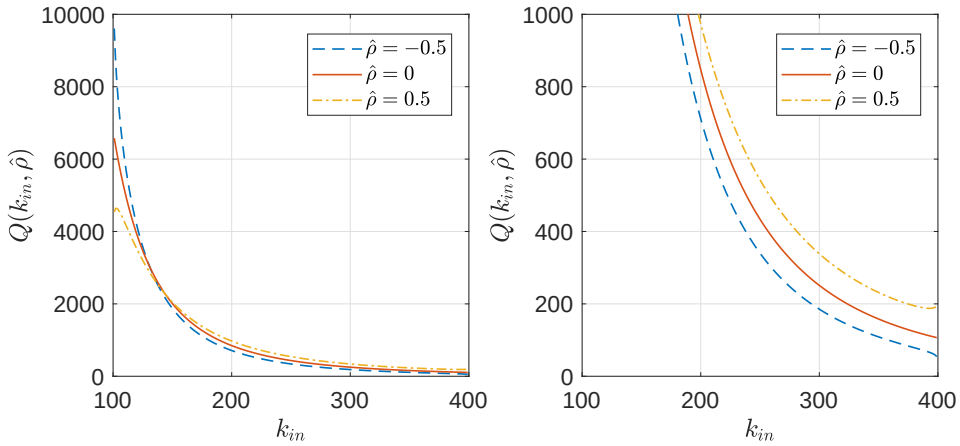

Figure 2.4: The function $Q(k_{in}, \hat{\rho})$ (Eqn. (2.3.15)) for different $\hat{\rho}$. The right panel is a zoom of the left panel. Parameters: $a = 100, b = 400, N = 2000$.

Note that the factor of N here cancels with that in the last term in (2.2.15), giving equations which do not explicitly depend on N. Examples of $Q(k'_{in}, \hat{\rho})$ for different $\hat{\rho}$ are shown in Fig. 2.4. We see that increasing $\hat{\rho}$ gives more weight to high in-degree nodes and less to low in-degree nodes and vice versa.

2.4 Model reduction to "virtual degrees"

We now turn to the issue of evaluating the sums over degrees in both (2.3.15) and (2.2.15). Although such sums are typically over only several hundred terms, it is possible to accurately evaluate them using many fewer terms, in analogy with Gaussian quadrature [Eng06]. (See Section 4.3 for more details.)

If we assume that the function $Q(k'_{in}, \hat{\rho})$ can be well approximated by a polynomial of degree $2n-1$, we can make use of the approximation

$$\sum_{k=a}^{b} f(k) \approx \sum_{i=1}^{n} w_i f(x_i) \tag{2.4.1}$$

where $n \ll b-a+1$, the number of terms in the original sum. We thus choose a number n, construct a set of n orthogonal polynomials $\{q_i(k)\}$, and write

$$Q(k'_{in}, \hat{\rho}) = N \sum_{j=1}^{n} w_j P(k'_{in}, k_j, \hat{\rho}) k_j \tag{2.4.2}$$

where k_j are the roots of the highest order polynomial $q_n(k)$ and w_j are the associated weights. In order to use the same approximation for the sum in (2.2.15), we consider only values of k_{in} equal to the k_j. As mentioned, these are typically *not* integers. We refer to them as "virtual degrees". Thus our model equations are

$$\frac{\partial b(k_j,t)}{\partial t} = \frac{-i(b(k_j,t)-1)^2}{2} + \frac{(b(k_j,t)+1)^2}{2}\left[-\Delta + i\eta_0 + \frac{iKk_j}{N\langle k\rangle^2}\sum_{j=1}^{n} w_j Q(k_j,\hat{\rho}) G(k_j,t)\right] \tag{2.4.3}$$

for $j = 1, \ldots n$. We are interested in fixed points of these equations, and how these fixed points and their stabilities change as parameters such as η_0 and $\hat{\rho}$ are varied. We use pseudo-arclength continuation [Lai14b, Gov00] to investigate this.

In order to calculate the mean frequency of the network we use the result that the frequency for neurons with in-degree k is [MPR15]

$$f(k) = \frac{1}{\pi}\mathrm{Re}\left(\frac{1-\bar{b}(k)}{1+\bar{b}(k)}\right), \tag{2.4.4}$$

where overline indicates complex conjugate, and then average over the network to obtain the mean frequency

$$\begin{aligned} f &= \frac{\sum_{k_{in}}\sum_{k_{out}} P(k_{in},k_{out},\hat{\rho}) f(k_{in})}{\sum_{k_{in}}\sum_{k_{out}} P(k_{in},k_{out},\hat{\rho})} \\ &= \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} w_i w_j P(k_i,k_j,\hat{\rho}) f(k_i)}{\sum_{i=1}^{n}\sum_{j=1}^{n} w_i w_j P(k_i,k_j,\hat{\rho})} \end{aligned} \tag{2.4.5}$$

(The normalisation is needed because even though the integral of the joint degree distribution over $[k_{in}, k_{out}]^2$ equals 1, the sum over the corresponding discrete grid does not.)

Typical convergence of a calculation of f with increasing n is shown in Fig. 2.5 for several sets of parameter values. We see rapid convergence and choose $n = 15$ for future

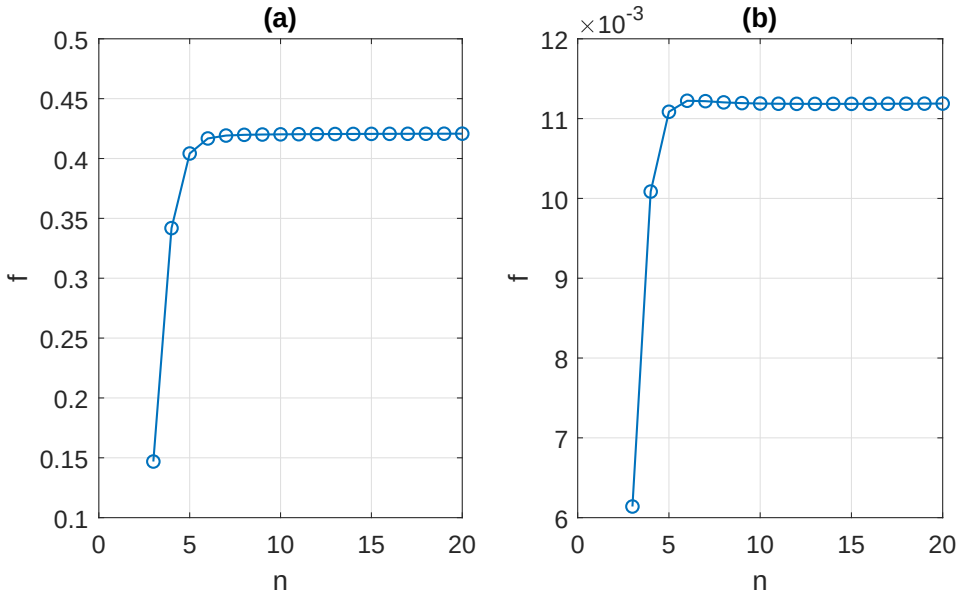

Figure 2.5: Mean frequency, f, as a function of n, the number of virtual degrees used to simulate the network. (a): $\hat{\rho} = -0.2, K = 1, \eta_0 = 0.5$. (b): $\hat{\rho} = 0.3, K = -0.1, \eta_0 = -0.5$. Other parameters: $a = 100, b = 400, \Delta = 0.05, N = 2000$.

calculations. (Calculations of the form shown in Figs. 2.6 and 2.8 were repeated using the full degree sequence from a to b, with essentially identical results.)

2.5 Results

2.5.1 Excitatory coupling

We first consider the case of excitatory coupling with $K > 0$. We expect a region of bistability for negative η_0, as seen in Fig. 2.6. We see that a decreasing ρ moves the curve to the right and vice versa ($\hat{\rho}$ was chosen to give these particular values of ρ). Following the saddle-node bifurcations as ρ is varied we obtain Fig. 2.7.

Given the influence of $\hat{\rho}$ (and thus ρ) on Q (see Fig. 2.4), this result is easy to understand. Neurons with high in-degree fire faster than those with low in-degree, and for positive ρ, high in-degree neurons contribute more to the sum in (2.4.3) than for negative ρ. Thus the total amount of "output" from neurons is higher for positive ρ and lower for negative ρ. Put another way, with positive ρ, neurons with high firing rate (due to high in-degree) are more likely to have a high out-degree, thus exciting more neurons than would otherwise be the case. Increasing ρ has the same qualitative effect as increasing the coupling strength K, as observed by [NFS+17].

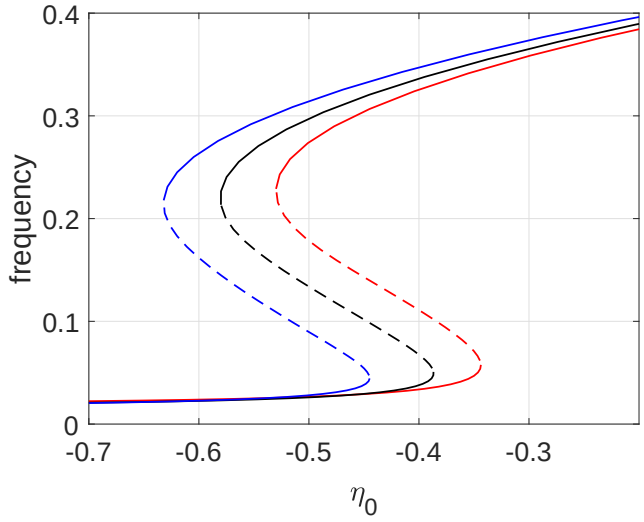

Figure 2.6: Mean frequency, f, versus intrinsic excitability η_0 for (left to right) $\rho = 0.5, 0$ and -0.5. Solid: stable, dashed: unstable. Parameters: $a = 100, b = 400, K = 1.5, \Delta = 0.05$.

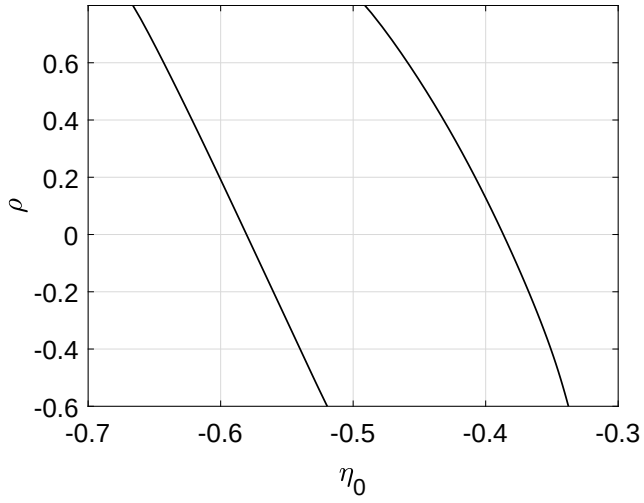

Figure 2.7: Continuation of the saddle-node bifurcations shown in Fig. 2.6. The network is bistable in the region between the curves. Parameters as in Fig. 2.6.

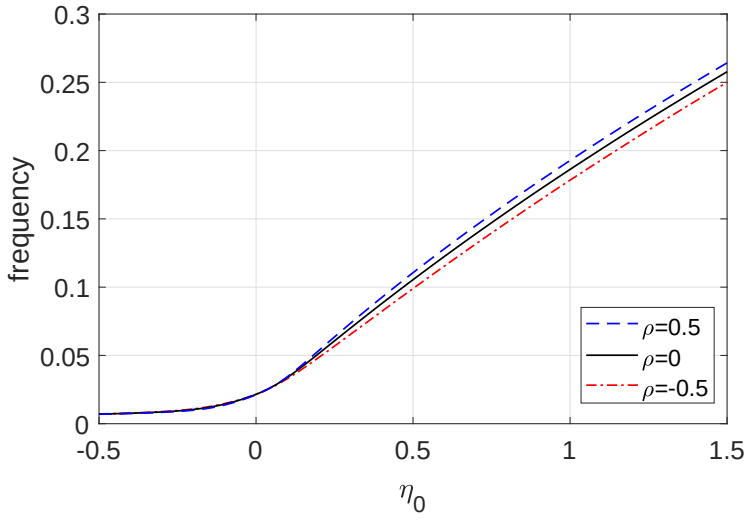

Figure 2.8: Mean frequency, f, versus η_0 for $\rho = -0.5, 0$ and 0.5; same colour code as in Fig. 2.6. All branches are stable. Parameters: $a = 100, b = 400, K = -1, \Delta = 0.05$.

2.5.2 Inhibitory coupling

Next we consider inhibitory coupling, with $K = -1$. Average network frequency f versus mean intrinsic excitability η_0 is shown in Fig. 2.8 for three different values of ρ. We see that increasing ρ slightly increases the frequency and vice versa. We can also understand this behaviour in a qualitative sense. For inhibitory coupling, neurons with high in-degree are not likely to be firing and can be ignored. When $\rho < 0$, neurons with low in-degree will have high out-degree, thus the amount of inhibitory "output" in the network is increased. For positive ρ, neurons with low in-degree will have low out-degree, thus they will inhibit fewer neurons than in the case of negative ρ, leading to a higher average firing rate.

We performed calculations corresponding to the results shown in Fig. 2.6 and Fig. 2.8 for networks of theta neurons and found qualitatively, and to a large extent quantitatively, the same behaviour as in those figures (results not shown).

2.6 Validation with a Morris-Lecar neuronal network

To verify the behaviour seen above in a network of theta neurons, we investigated a more realistic network of spiking neurons, in this case Morris-Lecar neurons. They feature calcium and potassium channels, a leak current and synaptic coupling. For each neuron there are three variables describing the membrane voltage V, the potassium gate activation n, and the synaptic output s. For the case of excitatory coupling the network equations for N

neurons are [TKY+06]

$$C\frac{dV_i}{dt} = g_L(V_L - V_i) + g_{Ca}m_\infty(V_i)(V_{Ca} - V_i) \tag{2.6.1}$$
$$+ g_K n_i(V_K - V_i)$$
$$+ I_0 + I_i + (V_{ex} - V_i)\frac{\epsilon}{N}\sum_{j=1}^{N} A_{ij}s_j$$
$$\frac{dn_i}{dt} = \frac{\lambda_0(w_\infty(V_i) - n_i)}{\tau_n(V_i)} \tag{2.6.2}$$
$$\tau\frac{ds_i}{dt} = m_\infty(V_i) - s_i \tag{2.6.3}$$

where

$$m_\infty(V) = 0.5(1 + \tanh\left[(V - V_1)/V_2\right]) \tag{2.6.4}$$
$$w_\infty(V) = 0.5(1 + \tanh\left[(V - V_3)/V_4\right]) \tag{2.6.5}$$
$$\tau_n(V) = \frac{1}{\cosh\left[(V - V_3)/(2V_4)\right]} \tag{2.6.6}$$

and $i \in \{1, \dots N\}$. The equilibrium potentials are given in mV and read $V_L = -60$, $V_{Ca} = 120$ and $V_K = -80$. The model assumes much faster dynamics for calcium currents, such that they are always in equilibrium with their activation function $m_\infty(V)$. Both, $m_\infty(V)$ and the steady-state potassium activation $w_\infty(V)$, are of similar shape and each contain two voltage parameters measured in mV. $V_1 = -1.2$ and $V_3 = 12$, respectively, determine the voltage of half activated gates, whereas $V_2 = 18$ and $V_4 = 17.4$ relate to the slope of the respective activation function. The maximum conductances $g_L = 2$, $g_K = 8$, and $g_{Ca} = 4$ are in $\mathrm{mS/cm^2}$. The overall coupling strength is $\epsilon = 5\mathrm{mS/cm^2}$. Further, we have the overall voltage independent activation time scale $\lambda_0 = 1/15\mathrm{msec}^{-1}$ and synaptic time scale $\tau = 100$. A neuron's electrical capacity is modelled with $C = 20\mu\mathrm{F/cm^2}$. The large reversal potential $V_{ex} = 120\mathrm{mV}$ ensures excitatory coupling. Time is measured in milliseconds and currents in $\mu\mathrm{A/cm^2}$. In the absence of coupling and heterogeneity a neuron undergoes a SNIC bifurcation as I_0 is increased through ~ 40. We have used synaptic coupling of the form in [EK90], but on a timescale τ rather than instantaneous as in that paper. The I_i are randomly chosen from a Lorentzian distribution with a zero mean value and a half-width at half-maximum of 0.05.

The network is created as follows, using the Gaussian copula of Sec. 2.3. For each $i \in \{1, \dots N\}$ let x_1 and x_2 be independently chosen from a unit normal distribution. Then x_1 and $y_1 = \hat{\rho}x_1 + \sqrt{1-\hat{\rho}^2}x_2$ both have unit normal distributions and covariance $\hat{\rho}$, i.e. are realisations of x and y in (2.3.2). We then set $k^i_{in} = C_k^{-1}(C(x_1))$ and $k^i_{out} = C_k^{-1}(C(y_1))$. These degrees each have distribution $p(k)$ but have correlation coefficient ρ, where ρ is

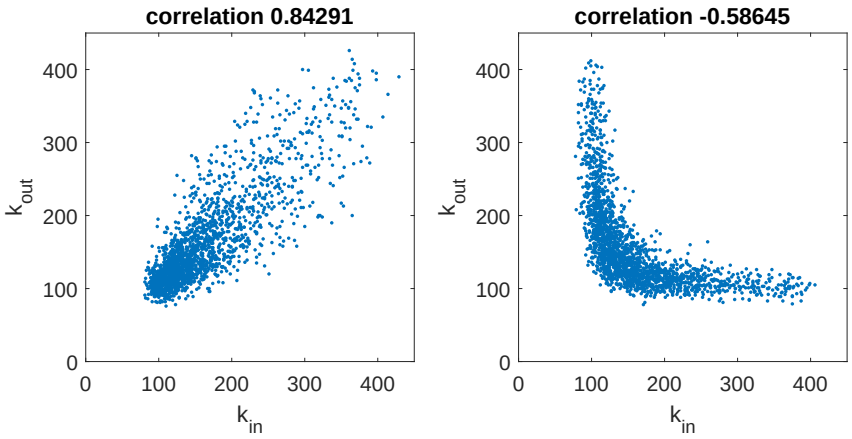

Figure 2.9: Degrees for a network whose generation is described in Sec. 2.6 for $\hat{\rho} = 0.9$ (left) and $\hat{\rho} = -0.9$ (right). Parameters: $N = 2000, a = 100, b = 400$.

determined by the value of $\hat{\rho}$ as shown in Fig. 2.3. We then create the connection from neuron j to neuron i (i.e. set $A_{ij} = 1$) with probability

$$\frac{k^i_{in} k^j_{out}}{N\langle k \rangle} \tag{2.6.7}$$

where $\langle k \rangle$ is the mean of the degrees, and $A_{ij} = 0$ otherwise (the Chung-Lu model [CL02]). Typical results for the network generation are shown in Fig. 2.9, and the measured correlations are given in the figure. The distributions of the resulting degrees no longer match the distributions of the k^i_{in} and k^i_{out}, but are close. We could have used the configuration model to avoid this problem [New03], but here we are only interested in qualitative results. Quasi-statically sweeping through I_0 for networks with three different values of ρ we obtain Fig. 2.10, in qualitative agreement with Fig. 2.6. In Fig. 2.6 there is a region of bistability for each value of ρ, and the region moves to lower average drive as ρ is increased. Since we cannot detect unstable states through simulation of (2.6.1)-(2.6.3), this bistability is manifested as jumps from low frequency to high frequency branches as I_0 is varied, as seen in Fig. 2.10.

For inhibitory coupling we replace $m_\infty(V_i)$ in (2.6.3) by $w_\infty(V_i)$, replace $V_{ex} - V_i$ in (2.6.1) by $V_K - V_i$, and choose $\epsilon = 10mS/cm^2$. Sweeping through I_0 for three different values of ρ we obtain Fig. 2.11, in qualitative agreement with Fig. 2.8.

2.7 Motifs

A number of authors have found that "motifs" (small sets of neurons connected in a specific way) do not occur in cortical networks in the proportions one would expect by

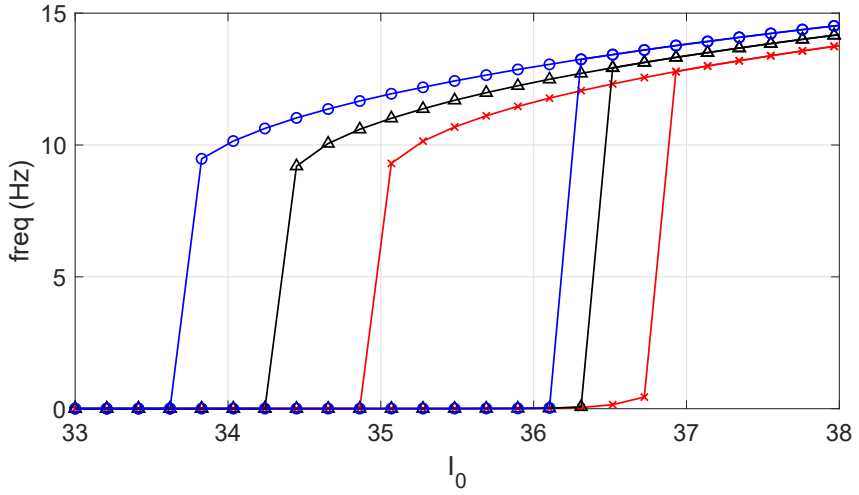

Figure 2.10: Mean frequency versus I_0 for a network of Morris-Lecar neurons. $N = 2000$. Red crosses: $\rho = -0.57$; black diamonds: $\rho = 0$; blue circles: $\rho = 0.85$. I_0 is quasi-statically increased and then decreased in all cases.

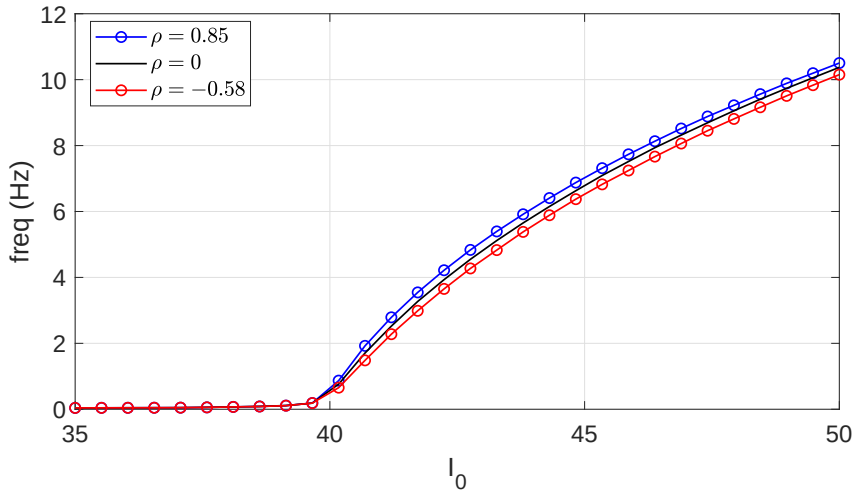

Figure 2.11: Mean frequency versus I_0 for networks of Morris-Lecar neurons with inhibitory coupling. $N = 2000$.

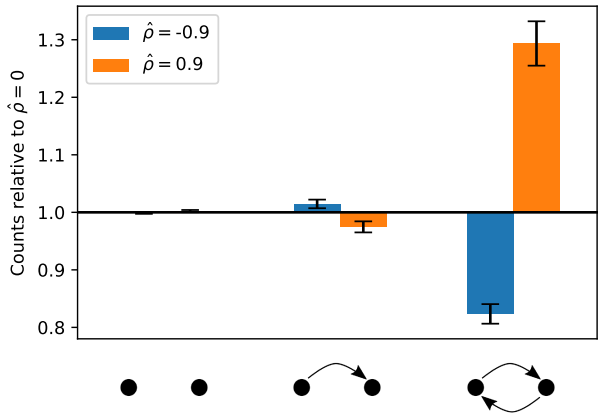

Figure 2.12: Relative counts of order-2 motifs. We generate three networks at a time with $\hat{\rho} \in [-0.9, 0, 0.9]$ to compute motif frequencies and repeat this process 100 times. Error bars indicate the standard deviation. Parameters are chosen as in Figure 2.9.

chance [SSR+05, PBM11]. Some theoretical results relating the presence or absence of certain motifs to network dynamics have been obtained [ZBNN11, HTJSB13, OLKD15]. For networks whose generation is described in Sec. 2.6 we counted the number of order-2 and order-3 motifs (involving two or three neurons respectively), for negative, zero and positive values of ρ. We compute the frequencies (amount) of order-2 motifs by counting the number of 0's, 1's and 2's in the upper triangular part of $(A + A^T)$, where A is the adjacency matrix and T means transposed. They refer to unconnected, unidirectional connected and reciprocal connected pairs of neurons, respectively. For all 13 connected order-3 motifs we used the software "acc-motif"[MMFDCa14]. The remaining three unconnected motifs have been counted by our own algorithm, that is looping through all neurons, we create for each a list of disconnected neurons and count among those order-2 motifs. The results are shown in Figs. 2.12 and 2.13, where counts are shown relative to the numbers found for $\rho = 0$.

In all motifs with at least one reciprocal connection between two neurons, we see that the number of motifs goes up with positive ρ and down with negative ρ. This can be understood in an intuitive way: suppose $0 < \rho$ and consider a neuron with a high out-degree. It is likely to connect to a neuron with a high in-degree. But this second neuron will also have a high out-degree and is therefore more likely to connect to the first neuron, which also has a high in-degree, forming a reciprocal connection. Similarly, suppose $\rho < 0$ and consider a neuron with high out-degree. It is likely to connect to a neuron with high in-degree but low out-degree. Thus it is unlikely that this second neuron will connect back to the first, which has a low in-degree.

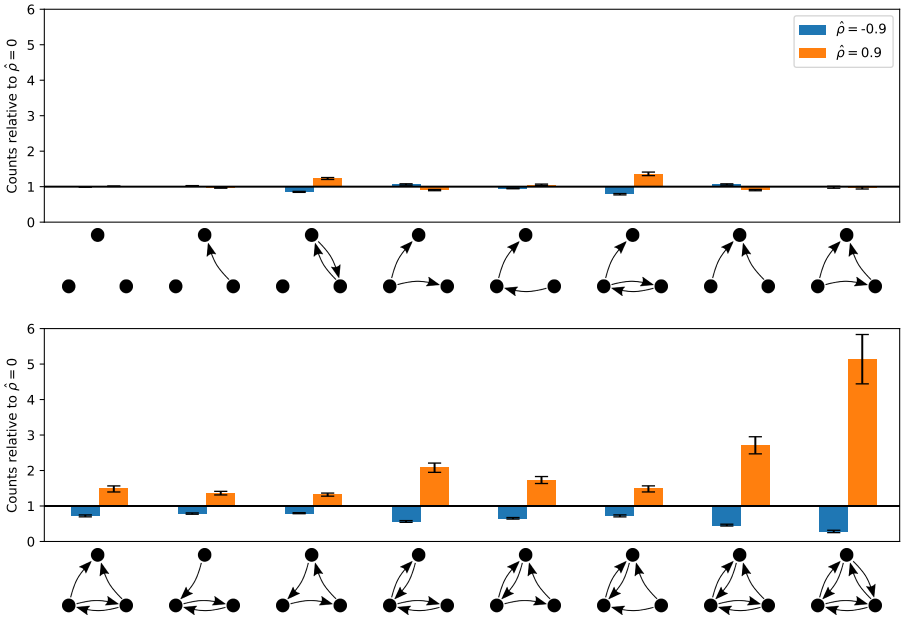

Figure 2.13: Relative counts of order-3 motifs.

2.8 Conclusion

We have investigated the effects of correlating the in- and out-degrees of spiking neurons in a structured network. We considered a large network of theta neurons, allowing us to exploit the analytical results previously derived by [CHC+17], which give dynamics for complex-valued order parameters, indexed by neurons with the same degrees. The states of interest are steady states of these dynamics, and by using a Gaussian copula we were able to analytically incorporate a parameter which controls the correlations between in- and out-degrees. Numerical continuation was then used to determine the effects of varying parameters, particularly the degree correlation. In order to reduce the computational cost we introduced the concept of "virtual degrees" allowing us to efficiently approximate sums with many terms by sums with fewer terms.

For an excitatory network we found that increasing degree correlations had a similar effect as increasing the overall strength of coupling between neurons, consistent with the findings of [NFS+17, VR19]. Our results are also consistent with those of [VHT13], who found that negative correlations stabilised the low firing rate state, as shown in Fig. 2.6. For inhibitory coupling we found that increasing degree correlations slightly increased the mean firing rate of the network. Both of these effects were reproduced in more realistic networks of Morris-Lecar spiking neurons.

We also measured the relative frequency of occurrence of order-2 and order-3 motifs as within-degree correlations were varied and found that in all motifs with at least one reciprocal connection between two neurons, the number of motifs is positively correlated with ρ. Several authors have linked motif statistics to synchrony within a network [HTJSB13, ZBNN11], however a link between motif statistics and firing rate, as observed here, seems yet to be developed.

We chose a Lorentzian distribution of the η_i in (2.2.1), as many others have done [OA08], in order to analytically evaluate an integral and derive (2.2.6). However, we repeated the calculations shown in Figs. 2.6, 2.8, 2.10 and 2.11 using a Gaussian distribution of the η_i and found the same qualitative behaviour (not shown). Regarding the parameter n governing the sharpness of the function $P_n(\theta)$, we repeated the calculations shown in Figs. 2.6 and 2.8 for $n = 5, \infty$ and obtained qualitatively the same results (not shown). We used a Gaussian copula to correlate in- and out-degrees due to its analytical form, but numerically investigated the scenarios shown in Figs. 2.6 and 2.8 for t copulas and Archimedean Clayton, Frank and Gumbel copulas and found the same qualitative behaviour (also not shown).

For simplicity we used the same truncated power law distribution for both in- and out-degrees. However, the use of a Gaussian copula for inducing correlations between degrees does not require them to be the same, so one could use the framework presented here to investigate the effects of varying degree distributions [Rox11], correlated or not.

We also only considered either excitatory or inhibitory networks, but it would be straightforward to generalise the techniques used here to the case of both types of neuron, with within-neuron degree correlations for either or both populations, though at the expense of increasing the number of parameters to investigate.

Chapter 3

Degree assortativity

Degree assortativity refers to the increased or decreased probability of connecting two neurons based on their in- or out-degrees, relative to what would be expected by chance. We investigate the effects of such assortativity in a network of theta neurons. The Ott/Antonsen ansatz is used to derive equations for the expected state of each neuron, and these equations are then coarse-grained in degree space. We generate families of effective connectivity matrices parametrised by the assortativity coefficient and use SVD decompositions of these to efficiently perform numerical bifurcation analysis of the coarse-grained equations. We find that of the four possible types of degree assortativity, two have no effect on the networks' dynamics, while the other two can have a significant effect.

The content of this section is an altered version of the publication [BML20].

3.1 Introduction

Our nervous system consists of a vast network of interconnected neurons. The network structure is dynamic and connections are formed or removed according to their usage. Much effort has been put into creating a brain atlas or *connectome*, which is a map of all neuronal interconnections. Given such a network there are many structural features and measures that one can use to characterise it, for instance betweenness, centrality, average path-length and clustering coefficient [New03].

Obtaining these measures in actual physiological systems is challenging to say the least; nevertheless, insights into intrinsic connectivity preferences of neurons were observed via their growth in culture [dSSSNL$^+$14, TGDD$^+$14]. Neurons with similar numbers of processes (e.g., synapses and dendrites) tend to establish links with each other – akin to socialites associating in groups and vice-versa. Such an assortativity, typically referred to as a positive assortativity, or a tendency of elements with similar properties to mutually connect,

emerges as a strong preference throughout the stages of the cultured neuron development. Furthermore, this preferential attachment between highly-connected neurons is suggested to fortify the neuronal network against disruption or damage [TGDD$^+$14]. Moreover, a similar positive assortativity is inferred in human central nervous systems as well [dSSSNL$^+$14] at both a structural and functional level, where a central "core" in the human cerebral cortex may be the basis for shaping overall brain dynamics [HCG$^+$08]. It seems that in no instance, however, is the directional flow of information (e.g., from upstream neuron via axon to synapse and downstream neuron) observed – either in culture or *in situ*.

As mentioned in the introduction (Sec. 1.1.1), assortativity in this context refers to the probability that a neuron with a given in- and out-degree connects to another neuron with a given in- and out-degree. If this probability is what one would expect by chance, given the neurons' degrees (and this is the case for all pairs of neurons), the network is referred to as neutrally assortative. If the probability is higher (lower) than one would expect by chance — for all pairs — the network is assortative (disassortative). Interchangeably, we will use the term positive assortativity (negative assortativity).

Assortativity has often been studied in undirected networks, where a node simply has a degree, rather than in- and out-degrees (the number of connections to and from a node, respectively) [RO14, New03, New02]. Since neurons form *directed* connections, there are four types of assortativity to consider [FFGP10]: between either the in- or out-degree of a presynaptic neuron, and either the in- or out-degree of a postsynaptic neuron (Figure 1.3).

We are aware of only a small number of previous studies in this area [SKSR15, KSR17, AGAP$^+$12, DFJT11]. Kähne et al. [KSR17] considered networks with equal in- and out-degrees and investigated degree assortativity, effectively correlating both in- and out-degrees of pre- and post-synaptic neurons. They mostly considered networks with discrete time and a Heaviside firing rate, that is a McCulloch-Pitts model [MP48]. They found that positive assortativity created new fixed points of the model dynamics. Schmeltzer et al. [SKSR15] also considered networks with equal in- and out-degrees and investigated degree assortativity. These authors considered leaky integrate-and-fire neurons and derived approximate self-consistency equations governing the steady state neuron firing rates. They found, among other things, that positive assortativity increased the firing rates of high-degree neurons and decreased that of low-degree ones. Positive assortativity also seemed to make the network more capable of sustained activity when the external input to the network was low. De Fransciscis et al. [DFJT11] considered assortative mixing of a field of binary neurons, or Hopfield networks. They concluded that assortativity of such simple model neurons exhibited associative memory (similar to bit fields of a magnetic storage medium), and robustly so in the presence of noise that negatively assortative networks failed to withstand. Avalos-Gaytan et al. [AGAP$^+$12] considered the effects of dynamic weightings between Kuramoto oscillators — effectively a dynamically evolving network — on assortativity. They observed that if

the strength of connections between oscillators increased when they were synchronised, a strong positive assortativity evolved in the network, suggesting a potential mechanism for the creation of assortative networks, as observed in cultured neurons mentioned above, and as we study here.

To briefly summarise our results, we find that only two out of the four types of degree assortativity have any influence on the network's dynamics: those when the in-degree of a presynaptic neuron is correlated with either the in- or out-degree of a postsynaptic neuron. Of these two, (in,in)-assortativity has a greater effect than (in,out)-assortativity. For both cases, negative assortativity widens the parameter range for which the network is bistable (for excitatory coupling) or undergoes oscillations in mean firing rate (for inhibitory coupling), and positive assortativity has the opposite effect.

Our work is similar in some respects to that of Restrepo and Ott [RO14] who considered degree assortativity in a network of Kuramoto-type phase oscillators. They found that for positive assortativity, as the strength of connections between oscillators was increased the network could undergo bifurcations leading to oscillations in the order parameter, in contrast to the usual scenario that occurs for no assortativity. However, their network was *undirected*, and thus there is only one type of degree assortativity possible.

The outline of this chapter is as follows. In Sec. 3.2 we present the model and then derive several approximate descriptions of its dynamics. In Sec. 3.3 we describe the method for creating networks with prescribed types of degree assortativity, and in Sec. 3.4 we discuss aspects of the numerical implementation of the reduced model. Results are given in Sec. 3.5 and we conclude with a discussion in Sec. 3.6.

3.2 A degree mean field featuring a derived assortativity function

We consider a network of N theta neurons:

$$\frac{d\theta_j}{dt} = 1 - \cos\theta_j + (1 + \cos\theta_j)(\eta_j + I_j) \tag{3.2.1}$$

for $j = 1, 2, \ldots N$ where

$$I_j = \frac{K}{\langle k \rangle} \sum_{n=1}^{N} A_{jn} P_q(\theta_n) \tag{3.2.2}$$

η_j is a constant current entering the jth neuron, randomly chosen from a distribution $g(\eta)$, K is strength of coupling, $\langle k \rangle$ is mean degree of the network, and the connectivity of the network is given by the adjacency matrix A, where $A_{jn} = 1$ if neuron n connects to neuron j, and zero otherwise. The connections within the network are either all excitatory (if $K > 0$) or inhibitory (if $K < 0$). Thus we do not consider the more realistic and general case of

a connected population of both excitatory and inhibitory neurons, although it would be possible using the framework below.

The theta neuron is the normal form of a Type I neuron which undergoes a saddle-node on an invariant circle bifurcation (SNIC) as the input current is increased through zero [Erm96, EK86]. A neuron is said to fire when θ increases through π, and the function

$$P_q(\theta) = a_q(1-\cos\theta)^q; \qquad q \in \{2,3,\dots\} \tag{3.2.3}$$

in (3.2.2) is meant to mimic the current pulse injected from neuron n to any postsynaptic neurons when neuron n fires. a_q is a normalisation constant such that $\int_0^{2\pi} P_q(\theta)d\theta = 2\pi$ independent of q.

3.2.1 An infinite ensemble

As a first step we consider an infinite ensemble of networks with the same connectivity, that is the same A_{jn}, but in each member of the ensemble, the value of η_j associated with the jth neuron is randomly chosen from the distribution $g(\eta)$ [BAO11]. Thus we expect a randomly chosen member of the ensemble to have values of η in the ranges

$$\begin{aligned}
\eta_1 &\in [\eta_1', \eta_1' + d\eta_1'] \\
\eta_2 &\in [\eta_2', \eta_2' + d\eta_2'] \\
&\vdots \\
\eta_N &\in [\eta_N', \eta_N' + d\eta_N']
\end{aligned} \tag{3.2.4}$$

with probability $g(\eta_1')g(\eta_2')\dots g(\eta_N')d\eta_1'd\eta_2'\dots d\eta_N'$. The state of this member of the ensemble is described by the probability density

$$f(\theta_1,\theta_2,\dots,\theta_N;\eta_1,\eta_2,\dots\eta_N;t) \tag{3.2.5}$$

which satisfies the continuity equation

$$\frac{\partial f}{\partial t} = -\sum_{j=1}^{N}\frac{\partial}{\partial\theta_j}\left\{\left[1-\cos\theta_j + (1+\cos\theta_j)\left(\eta_j + I_j\right)\right]f\right)\right\} \tag{3.2.6}$$

If we define the marginal distribution for the jth neuron as

$$f_j(\theta_j,\eta_j,t) = \int f(\theta_1,\theta_2,\dots\theta_N;\eta_1,\eta_2,\dots\eta_N;t)\prod_{k\neq j} d\theta_k d\eta_k \tag{3.2.7}$$

we can write

$$I_j(t) = \frac{K}{\langle k \rangle} \sum_{n=1}^{N} A_{jn} \int_{-\infty}^{\infty} \int_{0}^{2\pi} P_q(\theta_n) f_n(\theta_n, \eta_n, t) d\theta_n d\eta_n \tag{3.2.8}$$

where we have now evaluated I_j as an average over the ensemble rather than from a single realisation (as in (3.2.2)). This is reasonable in the limit of large networks [BAO11].

Multiplying (3.2.6) by $\prod_{k \neq j} d\theta_k d\eta_k$ and integrating we obtain

$$\frac{\partial f_j}{\partial t} = -\frac{\partial}{\partial \theta_j} \left\{ \left[1 - \cos\theta_j + (1 + \cos\theta_j)(\eta_j + I_j) \right] f_j \right\} \tag{3.2.9}$$

A network of theta neurons is known to be amenable to the use of the Ott/Antonsen ansatz [OA08, LBS13, Lai14a] so we write

$$f_j(\theta_j, \eta_j, t) = \frac{g(\eta_j)}{2\pi} \left[1 + \sum_{k=1}^{\infty} \{\alpha_j(\eta_j, t)\}^k e^{ik\theta_j} + \sum_{k=1}^{\infty} \{\bar{\alpha}_j(\eta_j, t)\}^k e^{-ik\theta_j} \right] \tag{3.2.10}$$

Appendix B contains the following derivation in more detail. The dependence on θ_j is written as a Fourier series where the kth coefficient is the kth power of a function α_j. Substituting this into (3.2.9) and (3.2.8) we find that α_j satisfies

$$\frac{\partial \alpha_j}{\partial t} = -i \left[\frac{\eta_j + I_j - 1}{2} + (1 + \eta_j + I_j)\alpha_j + \left(\frac{\eta_j + I_j - 1}{2} \right) \alpha_j^2 \right] \tag{3.2.11}$$

and

$$I_j(t) = \frac{K}{\langle k \rangle} \sum_{n=1}^{N} A_{jn} \int_{-\infty}^{\infty} H(\alpha_n(\eta_n, t); q) d\eta_n \tag{3.2.12}$$

where

$$H(\alpha; q) = a_q \left[C_0 + \sum_{n=1}^{q} C_n (\alpha^n + \bar{\alpha}^n) \right] \tag{3.2.13}$$

where an overbar indicates complex conjugate, and

$$C_n = \sum_{k=0}^{q} \sum_{m=0}^{k} \frac{\delta_{k-2m,n} q! (-1)^k}{2^k (q-k)! m! (k-m)!} \tag{3.2.14}$$

Assuming that g is a Lorentzian:

$$g(\eta) = \frac{\Delta/\pi}{(\eta - \eta_0)^2 + \Delta^2} \tag{3.2.15}$$

we can use contour integration to evaluate the integral in (3.2.12), and evaluating (3.2.11)

at the appropriate pole of g we obtain

$$\frac{dz_j}{dt} = \frac{-i(z_j-1)^2}{2} + \frac{(z_j+1)^2}{2}\left[-\Delta + i\eta_0 + iJ_j\right] \tag{3.2.16}$$

where

$$J_j = \frac{K}{\langle k\rangle}\sum_{n=1}^{N} A_{jn}H(z_n;q) \tag{3.2.17}$$

and $z_j = \langle e^{i\theta_j}\rangle$, where the expected value is taken over the ensemble.

Now (3.2.16) is a set of N coupled complex ODEs, so we have not simplified the original network (3.2.1) in the sense of decreasing the number of equations to solve. However, the states of interest are often *fixed points* of (3.2.16) (but not of (3.2.1)), and can thus be found and followed as parameters are varied. At this point the network we consider, with connectivity given by A, is arbitrary. If A was a circulant matrix, for example, this would represent a network of neurons on a circle, where the strength of coupling between neurons depends only on the distance between them [Lai14a].

3.2.2 Lumping by degree

The next step is to assume that for a large enough network, the dynamics of neurons with the same degrees will behave similarly [CHC+17]. Such an assumption has been made a number of times in the past [KSR17, Ich04, RO14]. We thus associate with each neuron the degree vector $\boldsymbol{k} = (k^{\text{in}}, k^{\text{out}})$ and assume that the values of z for all neurons with a given $\boldsymbol{k}$ are similar. There are $N_{\boldsymbol{k}} = N_{k^{\text{in}}}N_{k^{\text{out}}}$ distinct degrees where $N_{k^{\text{in}}}$ and $N_{k^{\text{out}}}$ are the number of distinct in- and out-degrees, respectively. We define b_s to be the order parameter for neurons with degree $\boldsymbol{k}_s$, where $s \in [1, N_{\boldsymbol{k}}]$, and now derive equations for the evolution of the b_s.

Let $\boldsymbol{z}$ be the vector of ensemble states z_j, where $j \in [1, N]$ and the degree index of neuron j be $d(j)$, such that $\boldsymbol{k}_{d(j)}$ is its degree. We assume that for all neurons with the same degree $\boldsymbol{k}_{d(j)} = \boldsymbol{k}_s$ the ensemble state z_j is similar in sufficiently large networks and thus we only care about the mean value $\langle z_j\rangle_{d(j)=s} = b_s$ with $s \in [1, N_{\boldsymbol{k}}]$. We say that degree $\boldsymbol{k}_s$ occurs h_s times and thus write

$$\boldsymbol{b} = C\boldsymbol{z}, \tag{3.2.18}$$

where the $N_{\boldsymbol{k}} \times N$ matrix C has h_s entries in row s, each of value $1/h_s$, at positions j where $d(j) = s$ and zeros elsewhere, that is $C_{sj} = \delta_{s,d(j)}/h_s$ with δ being the Kronecker delta.

To find the time derivative of $\boldsymbol{b}$ we need to express $\boldsymbol{z}$ in terms of $\boldsymbol{b}$, which we do with an

$N \times N_{\boldsymbol{k}}$ matrix B which assigns to z_j the corresponding b_s value, such that

$$\boldsymbol{z} = B\boldsymbol{b}, \tag{3.2.19}$$

with components $B_{js} = \delta_{d(j),s}$. Note that $CB = I_{N_{\boldsymbol{k}}}$, the $N_{\boldsymbol{k}} \times N_{\boldsymbol{k}}$ identity matrix. Differentiating (3.2.18) with respect to time, inserting (3.2.16) into this and writing $\boldsymbol{z}$ in terms of $\boldsymbol{b}$ using (3.2.19) we obtain

$$\begin{aligned}\dot{b_s} = \underbrace{\sum_{j=1}^{N} C_{sj} \left[-i\frac{\left(\sum_{t=1}^{N_{\boldsymbol{k}}} B_{jt} b_t - 1\right)^2}{2} + i\frac{\left(\sum_{t=1}^{N_{\boldsymbol{k}}} B_{jt} b_t + 1\right)^2}{2} (\eta_0 + i\Delta) \right]}_{\text{local}} \\ + \underbrace{\sum_{j=1}^{N} C_{sj} \left[i\frac{\left(\sum_{t=1}^{N_{\boldsymbol{k}}} B_{jt} b_t + 1\right)^2}{2} J_j \right]}_{\text{non-local}}.\end{aligned} \tag{3.2.20}$$

Considering that for all t there is only a single non-zero entry B_{jt}, equal to 1, the identity

$$\left(\sum_{t=1}^{N_{\boldsymbol{k}}} \underbrace{B_{jt}}_{=\delta_{d(j),t}} b_t \right)^n = b_{d(j)}^n \tag{3.2.21}$$

holds for any power n. Further we find that

$$\sum_{j=1}^{N} \underbrace{C_{sj}}_{=1/h_s \cdot \delta_{s,d(j)}} b_{d(j)} = b_s. \tag{3.2.22}$$

Thus, the local term in (3.2.20) is

$$\dot{b_s}^{\text{ local}} = -i\frac{(b_s - 1)^2}{2} + i\frac{(b_s + 1)^2}{2} (\eta_0 + i\Delta). \tag{3.2.23}$$

For the non-local term we write

$$\begin{aligned}\dot{b_s}^{\text{ non-local}} &= \sum_{j=1}^{N} \frac{1}{h_s} \delta_{s,d(j)} i\frac{\left(b_{d(j)} + 1\right)^2}{2} J_j \\ &= i\frac{(b_s + 1)^2}{2} \underbrace{\sum_{j=1}^{N} \frac{1}{h_s} \delta_{s,d(j)} J_j}_{=\sum_{j=1}^{N} C_{sj} J_j = \tilde{J}_s},\end{aligned} \tag{3.2.24}$$

where $\tilde{J}_s$ describes the synaptic current of the ensemble equations averaged over nodes sharing the same degree $\boldsymbol{k}_s$. The identity (3.2.21) also applies to (3.2.17), so that

$$H(z_n;q) = H\left(\sum_{t=1}^{N_{\boldsymbol{k}}} B_{nt} b_t; q\right) = \sum_{t=1}^{N_{\boldsymbol{k}}} B_{nt} H(b_t;q) \tag{3.2.25}$$

and the current can be written as

$$\begin{aligned}\tilde{J}_s &= \sum_{j=1}^{N} C_{sj} \frac{K}{\langle k\rangle} \sum_{n=1}^{N} A_{jn} \sum_{t=1}^{N_{\boldsymbol{k}}} B_{nt} H(b_t;q) \\ &= \frac{K}{\langle k\rangle} \sum_{t=1}^{N_{\boldsymbol{k}}} \underbrace{\sum_{j=1}^{N} \sum_{n=1}^{N} C_{sj} A_{jn} B_{nt}}_{E_{st}} H(b_t;q)\end{aligned} \tag{3.2.26}$$

The effective connectivity between neurons with different degrees is therefore expressed in the matrix $E = CAB$ and we end up with equations governing the b_s:

$$\frac{db_s}{dt} = \frac{-i(b_s-1)^2}{2} + \frac{(b_s+1)^2}{2}\left[-\Delta + i\eta_0 + i\tilde{J}_s\right] \tag{3.2.27}$$

where

$$\tilde{J}_s = \frac{K}{\langle k\rangle} \sum_{t=1}^{N_{\boldsymbol{k}}} E_{st} H(b_t;q) \tag{3.2.28}$$

These equations are of the same form as (3.2.16)-(3.2.17) except that A has been replaced by E. Note that the connectivity matrix A is completely general; we have only assumed that neurons with the same degrees behave in the same way. We are not aware of a derivation of this form being previously presented.

3.3 Network assembly

We are interested in the effects of degree assortativity on the dynamics of the network of neurons. We will choose a default network with no assortativity and then introduce one of the four types of assortativity and investigate the changes in the network's dynamics. Our default network is of size $N = 5000$ neurons where in- and out-degrees k for each neuron are independently drawn from the interval $[750, 2000]$ with probability $P(k) \sim k^{-3}$ (i.e. a power law, as found in [ECC+05] and used in [CHC+17]). We create networks using the configuration model [New03], then modify them using algorithms which introduce assortativity and then remove multiple connections between nodes (or multi-edges) (described in Appendix C). We choose as our default parameters $\eta_0 = -2, \Delta = 0.1, K = 3$, for which a default network approaches a stable fixed point. The sharpness of the synaptic pulse

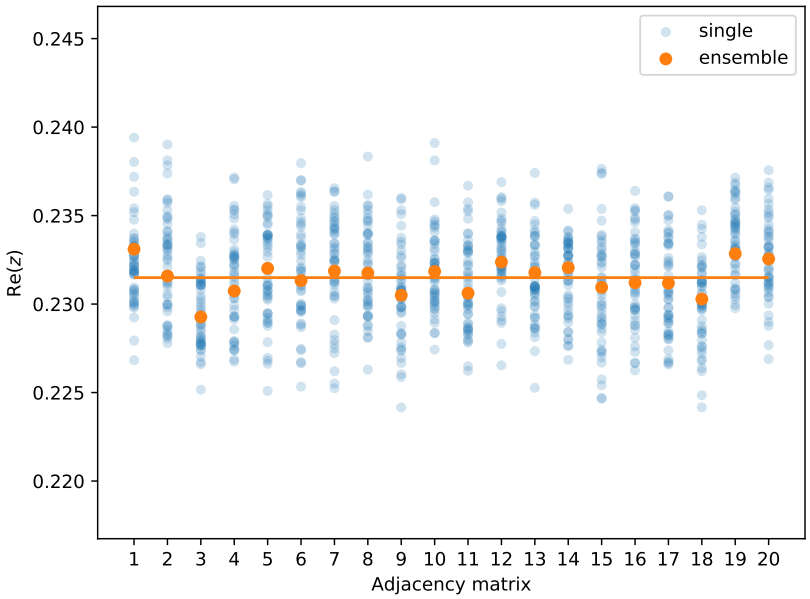

Figure 3.1: Orange circles: steady state of (3.2.16)-(3.2.17) for 20 different default networks. Blue circles: results from 50 different realisations of the η_i for (3.2.1)-(3.2.2), for each network. Parameters: $\eta_0 = -2, \Delta = 0.1, K = 3$. The orange line marks the ensemble mean value.

function is set to $q = 2$ for all simulations.

We first check the validity of averaging over an infinite ensemble. We assemble 20 different default networks and for each, run (3.2.16)-(3.2.17) to a steady state and calculate the order parameter z, the mean of $B\boldsymbol{b}$. The real part of z is plotted in orange in Fig. 3.1. For each of these networks we then generated 50 realisations of the η_i's and ran (3.2.1)-(3.2.2) for long enough that transients had decayed, and then measured the corresponding order parameter for the network of individual neurons

$$R = \frac{1}{N}\sum_{j=1}^{N} e^{i\theta_j} \tag{3.3.1}$$

and plotted its real part in blue in Fig. 3.1. Note that the orange circles always lie well within the range of values shown in blue. The fact that deviations within the 50 realisations are small relative to the value obtained by averaging over an infinite ensemble provide evidence for the validity of this approach, at least for these parameter values.

We also investigate the influence of multi-connections (i.e. more than one connection) between neurons on the network dynamics. The configuration model creates a network in which the neuron degrees are exactly those specified by the choice from the appropriate

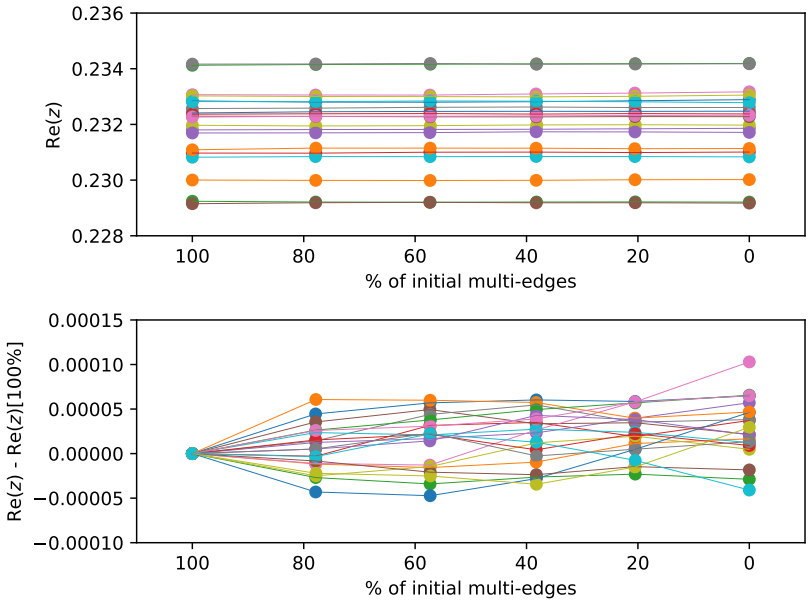

Figure 3.2: 20 different default adjacency matrices (indicated by different colours) are created, and then multi-edges are systematically removed. Top: real part of the order parameter at steady state. Bottom: difference of real part of z from that obtained before multi-edges are removed. There is no significant change nor trend observable while removing multi-edges. Parameters: $\eta_0 = -2, \Delta = 0.1, K = 3$.

distribution, but typically results in both self-connections and multiple connections between neurons. We have an algorithm (Appendix C) for systematically removing such connections while preserving the degrees, and found that removing such edges has no significant effect. We show the effect of removing such edges in Fig. 3.2. We create 20 default adjacency matrices and run (3.2.16)-(3.2.17) to a steady state, keeping 100% of all multi-edges. We then remove some fraction of initial multi-edges and repeat the process, continuing until no multi-edges remain. The real part of the order parameter for all cases is shown in Fig. 3.2, and we see that (for these parameters) variations between the default matrices are greater than those caused by removing all multi-edges. However, in our simulations we use simple graphs without multi-edges.

A novel technique to introduce multi-edges has been developed by the authors of [MBL20], which allows further investigation in this field.

3.3.1 Assortativity

For a given matrix A we can measure its assortativity by calculating the four Pearson correlation coefficients $r(\alpha, \beta)$ with $\alpha, \beta \in [\text{in}, \text{out}]$ which read

$$r(\alpha,\beta) = \frac{\sum_{e=1}^{N_e} ({}^s k_e^\alpha - \langle {}^s k^\alpha \rangle)({}^r k_e^\beta - \langle {}^r k^\beta \rangle)}{\sqrt{\sum_{e=1}^{N_e} ({}^s k_e^\alpha - \langle {}^s k^\alpha \rangle)^2} \sqrt{\sum_{e=1}^{N_e} ({}^r k_e^\beta - \langle {}^r k^\beta \rangle)^2}} \tag{3.3.2}$$

where

$$\langle {}^s k^\alpha \rangle = \frac{1}{N_e} \sum_{e=1}^{N_e} {}^s k_e^\alpha \qquad \text{and} \qquad \langle {}^r k^\beta \rangle = \frac{1}{N_e} \sum_{e=1}^{N_e} {}^r k_e^\beta, \tag{3.3.3}$$

N_e being the number of connections and the leading superscript s or r refers to the sending or receiving neuron of the respective edge. For example the sending node's in-degree of the second edge would be ${}^s k_2^{in}$. Note that there are four mean values to compute.

We introduce assortativity by randomly choosing two edges and swapping postsynaptic neurons when doing so would increase the target assortativity coefficient [SKSR15]. An edge (i, j) is directed from neuron j to neuron i. In order to know whether the pair (i, j) and (h, l) should be rewired or left untouched, we compare their contribution to the covariance in the numerator of (3.3.2):

$$\begin{aligned} c_\parallel &= c((i,j),(h,l)) \\ &= \left(k_j^\alpha - \langle {}^s k^\alpha \rangle\right)\left(k_i^\beta - \langle {}^r k^\beta \rangle\right) + \left(k_l^\alpha - \langle {}^s k^\alpha \rangle\right)\left(k_h^\beta - \langle {}^r k^\beta \rangle\right); \end{aligned} \tag{3.3.4}$$

$$\begin{aligned} c_\times &= c((i,l),(h,j)) \\ &= \left(k_l^\alpha - \langle {}^s k^\alpha \rangle\right)\left(k_i^\beta - \langle {}^r k^\beta \rangle\right) + \left(k_j^\alpha - \langle {}^s k^\alpha \rangle\right)\left(k_h^\beta - \langle {}^r k^\beta \rangle\right). \end{aligned} \tag{3.3.5}$$

If $c_\times > c_\parallel$ we replace the edges (i, j) and (h, l) by (i, l) and (h, j), respectively, otherwise we do not, and continue by randomly choosing another pair of edges. Algorithm 3 (see Appendix C) demonstrates a scheme for reaching a certain target assortativity coefficient.

We investigate the effects of different types of assortativity (see Fig 1.3) in isolation. We thus need a family of networks parametrised by the relevant assortativity coefficient. Algorithm 3 is used to create a network with a specific value of one of the assortativity coefficients, but especially for high values of assortativity it may be that in doing so a small amount of assortativity of a type other than the intended one is introduced. Accordingly, it may be necessary to examine all types of assortativity and apply the mixing scheme to reduce other types back to zero, and then (if necessary) push the relevant value of assortativity back to its target value. We do multiple iterations of these mixing rounds until all assortativities are at their target values (which may be 0) within a range of ± 0.005. We use Algorithm 3

with a range of target assortativities r, and for each value, store the connectivity matrix A and thus form the parametrised family $E(r)$. We do this for the four types of assortativity.

We have chosen to use the configuration model (Appendix A.1) to create networks with given degree sequences and then introduced assortativity by swapping edges. By contrast, another common adjacency network assembly technique, that of Chung and Lu [CL02] together with an analytical expression for assortativity (as in [CHC$^+$17]), proved inadequate. We found that the latter approach significantly alters the degree distribution for large assortativity, whereas the configuration model combined with our mixing algorithm does not change degrees at all. More details on this topic can be found in the Appendix A.2. For our default network this approach allows us to introduce assortativity of any one kind up to $r = \pm 0.5$.

3.4 Implementation

For networks of the size we investigate it is impractical to consider each distinct in- and out-degree (because E will be very large and sparse). Due to the smoothness of the degree dependency of $b(\boldsymbol{k})$ we coarse-grain in degree space by introducing "degree clusters" — lumping all nodes with a range of degrees into a group with dynamics described by a single variable. Let there be $N_{c^{\text{in}}}$ clusters in in-degree and $N_{c^{\text{out}}}$ clusters in out-degree, with a total of $N_{\boldsymbol{c}} = N_{c^{\text{in}}} \cdot N_{c^{\text{out}}}$ degree clusters. The matrix C then is an $N_{\boldsymbol{c}} \times N$ matrix and constructed as previously, except that $d(j)$ is not the degree index of neuron j, but the cluster index and s is the cluster index running from 1 to $N_{\boldsymbol{c}}$. Similarly for the matrix B. There are multiple options for how to combine degrees into a cluster. We can split the degree space evenly and assign a cluster index to each interval. However, with this approach, depending on the degree distribution, some of the clusters may be empty or hardly filled, resulting in poor statistics. To overcome this issue, the cumulative sum of in- and out-degree distribution can be used to map degrees to cluster indices. Thus, clusters are more evenly filled and at the same time regions of degree space with high degree probability are more finely sampled. The dynamical equations (3.2.27)-(3.2.28) are equally valid for describing degree cluster dynamics with $s, t \in [1, N_{\boldsymbol{c}}]$ and $E = CAB$, where C and B are cluster versions of their previous definitions.

To check the effect of varying the number of clusters we generate 20 default matrices and then generate the corresponding matrix E with varying numbers of clusters ($N_{c^{\text{in}}}$ and $N_{c^{\text{out}}}$ are equal), then run (3.2.27)-(3.2.28) to a steady state and plot the real part of z in Fig. 3.3. We see that the order parameter is well approximated using as little as about $N_{c^{\text{in}}} = N_{c^{\text{out}}} = 10$ degree clusters. Beyond that, fluctuations between different network realisations exceed the error introduced by clustering. In our simulations we stick to the choice of 10 degree clusters per in- and out-degree space.

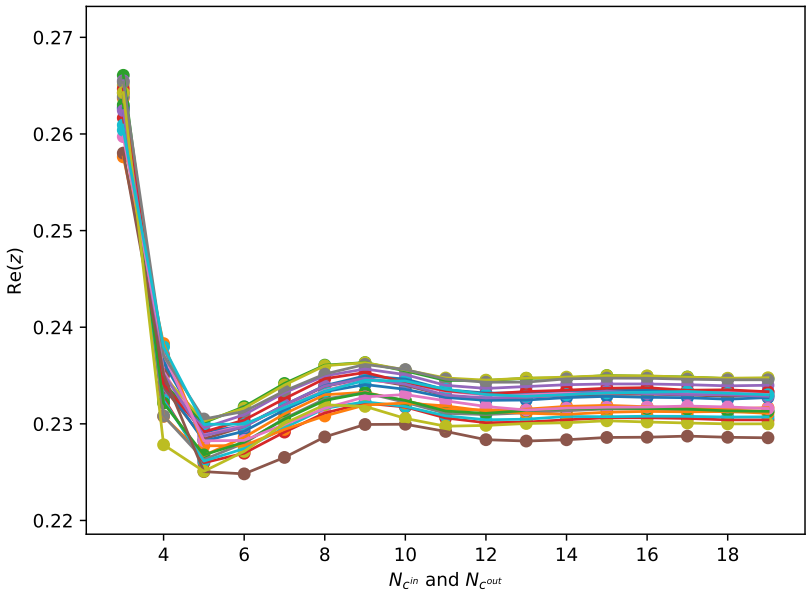

Figure 3.3: Real part of z at steady state for 20 different default adjacency matrices (indicated by different colours), as the number of clusters in degree space is varied. Parameters: $\eta_0 = -2, \Delta = 0.1, K = 3$.

Having performed this clustering, we find that it is possible to represent E using a low-rank approximation, calculated using singular value decomposition (Sec. 4.1). Thus for a fixed r we have

$$E = USV^T \tag{3.4.1}$$

where S is a diagonal matrix with decreasing entries, called singular values, and U and V are unitary matrices. In Fig. 3.4 we plot the largest 6 singular values of E as a function of the assortativity coefficient, for the 4 types of assortativity. Even for large $|r|$ the singular values decay very quickly, thus a low-rank approximation is possible. We choose a rank-3 approximation, so approximate E by

$$E(r) \approx \left[\begin{array}{c|c|c} u_1(r) & u_2(r) & u_3(r) \end{array}\right] \begin{bmatrix} s_1(r) & 0 & 0 \\ 0 & s_2(r) & 0 \\ 0 & 0 & s_3(r) \end{bmatrix} \left[\begin{array}{c} v_1^T(r) \\ \hline v_2^T(r) \\ \hline v_3^T(r) \end{array}\right] \tag{3.4.2}$$

where u_i is the ith column of U, similarly for v_i and V, and s_i is the ith singular value. We

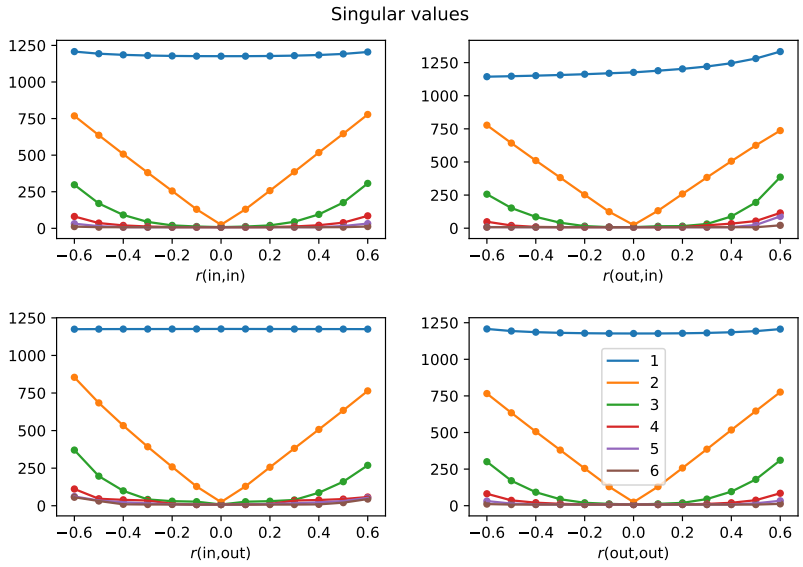

Figure 3.4: Six largest singular values of the SVD decomposition of E as a function of assortativity coefficient, for 4 types of assortativity.

have such a decomposition at discrete values of r and use cubic interpolation to evaluate $E(r)$ for any r. This decomposition means that the multiplication in (3.2.28) can be evaluated quickly using 3 columns of U and V rather than the full $N_{\boldsymbol{c}} \times N_{\boldsymbol{c}}$ matrix E.

We note that the components for the approximation of $E(r)$ are calculated once and then stored, making it very easy to systematically investigate the effects of varying any of the parameters η_0, Δ, K and q (governing the sharpness of the pulse function (3.2.3)).

3.5 Results

3.5.1 Excitatory coupling

We take $K = 3$ to model a network with only excitatory connections. To study the dynamical effect of assortativity we generate positive and negative ($r = \pm 0.2$) assortative networks of the four possible kinds and follow fixed points of (3.2.27)-(3.2.28) as a function of η_0, and compare results with a neutral ($r = 0$) network. We use pseudo-arc-length continuation [Lai14b, Gov00].

To calculate the mean frequency over the network we evaluate $\boldsymbol{z} = B\boldsymbol{b}$ and then use the result that if the order parameter at a node is z, then the frequency of neurons at that node is [Lai15, MPR15]

$$\frac{1}{\pi}\mathrm{Re}\left(\frac{1-\bar{z}}{1+\bar{z}}\right) \tag{3.5.1}$$

Averaging these gives the mean frequency.

Results are shown in Figure 3.5, where we see quite similar behaviour in each case: apart from a bistable region containing two stable and one unstable fixed points, there is only a single stable fixed point present. Further, the two assortativity types (out,in) and (out,out) apparently do not affect the dynamics, whereas the saddle-node bifurcations marking the edges of the bistable region move slightly for (in,out) and significantly for (in,in) assortativity. Following the saddle-node bifurcations for the latter two cases we find the results shown in Figure 3.6. We have performed similar calculations for different networks with the same values of assortativity and found similar results.

3.5.2 Inhibitory coupling

We choose $K = -3$ to model a network with only inhibitory coupling. Again, we numerically continue fixed points for zero, positive and negative assortativity ($r = 0, \pm 0.2$) as η_0 is varied and obtain the curves shown in Figure 3.7. Consider the lower left plot. For large η_0 the system has a single stable fixed point which undergoes a supercritical Hopf bifurcation as η_0 is decreased, creating a stable periodic orbit. This periodic orbit is destroyed in a saddle-node bifurcation on an invariant circle (SNIC) bifurcation at lower η_0, forcing the oscillations to stop. Decreasing η_0 further, two unstable fixed points are destroyed in a saddle-node bifurcation. In contrast with the case of excitatory coupling, oscillations in the average firing rate are seen. These can be thought of as partial synchrony, since some fraction of neurons in the network have the same period and fire at similar times to cause this behaviour. The period of this macroscopic oscillation tends to infinity as the SNIC bifurcation is approached, as shown in the inset of the lower left panel in Fig. 3.7.

As in the excitatory case, we see that assortativities of type (out,in) and (out,out) have no influence on the dynamics in this scenario. However, type (in,out) does have a small effect, slightly moving bifurcation points (top right panel in Fig. 3.7). Type (in,in) has the strongest effect, resulting in a qualitative change in the bifurcation scenario for large enough assortativity: there is a region of bistability between either two fixed points or a fixed point and a periodic orbit. This is best understood by following the bifurcations in the top panels of Fig. 3.7 as r is varied, as shown in Figure 3.8. There is one fixed point in regions A, B and D, and three in region C. For (in,out) assortativity there is a stable periodic orbit in region B and never any bistability.

We now describe the case for (in,in) assortativity. For negative and zero r the scenario is the same as for the other three types, but as r is increased there is a Takens-Bogdanov bifurcation where regions C,D,E and F meet, leading to the creation of a curve of homoclinic bifurcations, which is destroyed at another codimension-two point where there is a homoclinic connection to a non-hyperbolic fixed point [CL90]. There are stable oscillations in region E, created or destroyed in supercritical Hopf or homoclinic bifurcations. In region

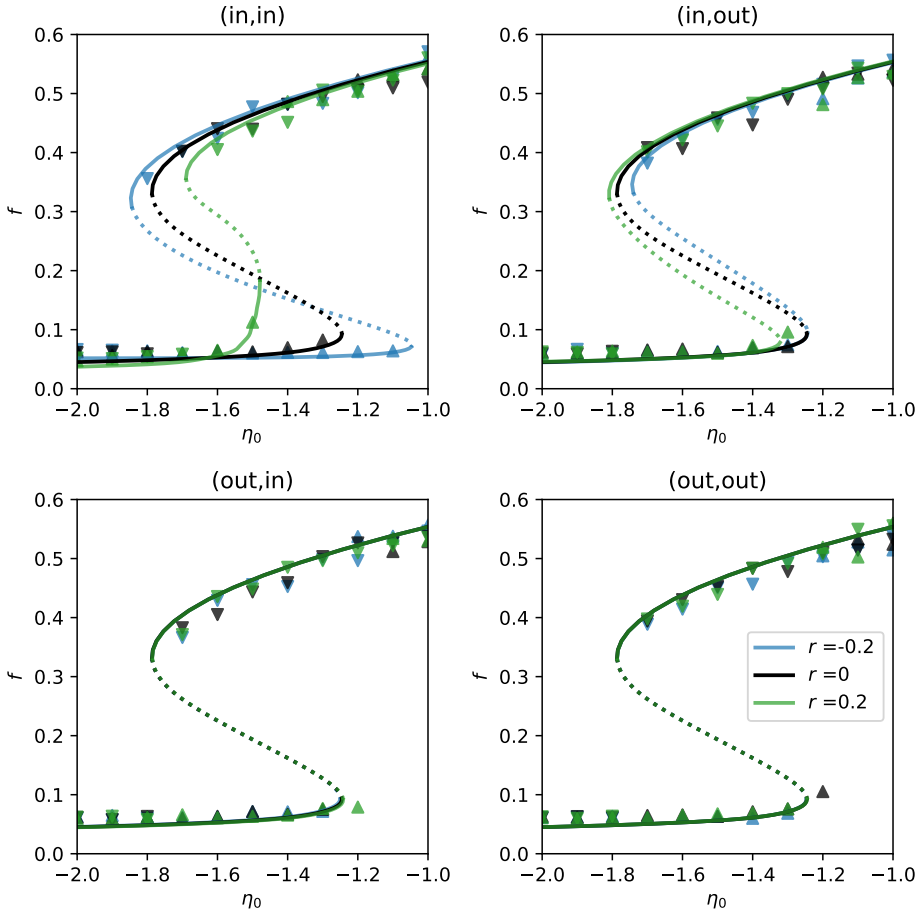

Figure 3.5: Average firing rate at fixed points of (3.2.27)-(3.2.28) as a function of η_0, for the 4 types of assortativity. For each type of assortativity curves are plotted for $r = 0$ (black), $r = -0.2$ (blue) and $r = 0.2$ (green). Solid lines indicate stable and dashed lines unstable fixed points. Triangular data points are computed using the full theta neuron model. Parameters: $K = 3, \Delta = 0.1$.

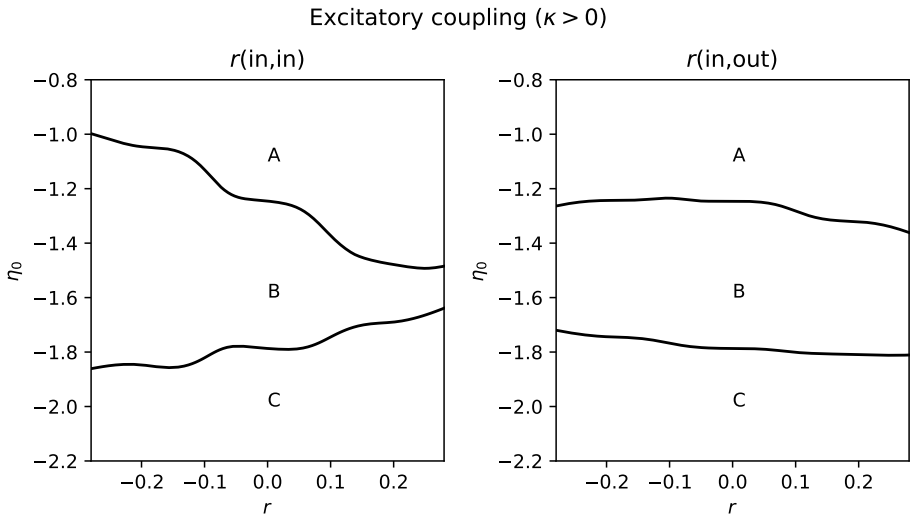

Figure 3.6: Continuation of the saddle-node bifurcations seen in the upper two panels of Fig. 3.5 as r is varied. Curves in Figure 3.5 correspond to vertical slices at $r = 0, \pm 0.2$. The network is bistable in region B and has a single stable fixed point in regions A and C.

F there is bistability between two fixed points.

3.6 Discussion

We investigated the effects of degree assortativity on the dynamics of a network of theta neurons. We used the Ott/Antonsen ansatz to derive evolution equations for an order parameter associated with each neuron, and then coarse-grained by degree and then degree cluster, obtaining a relatively small number of coupled ODEs, whose dynamics as parameters varied could be investigated using numerical continuation. We found that degree assortativity involving the out-degree of the sending neuron, that is (out,in) and (out,out), has no effect on the networks' dynamics. Further, (in,out) assortativity moves bifurcations slightly, but does not lead to substantial differences in dynamical behaviour. The most significant effects were caused by creating correlation between in-degrees of the sending and receiving neurons. For our excitatorially coupled example, positive (in,in) assortativity narrows the bistable region, whereas negative assortativity widens it (see Fig. 3.6). In the inhibitory case introducing negative assortativity increased the amplitude of network oscillations and extended their range to slightly larger η_0. On the contrary, positive (in,in) assortativity in this network has an opposite effect and eventually stops oscillations (see Fig. 3.8).

The most similar work to ours is that of [CHC+17]. These authors also considered a network of the form (3.2.1)-(3.2.2) and by assuming that the dynamics depend on only a

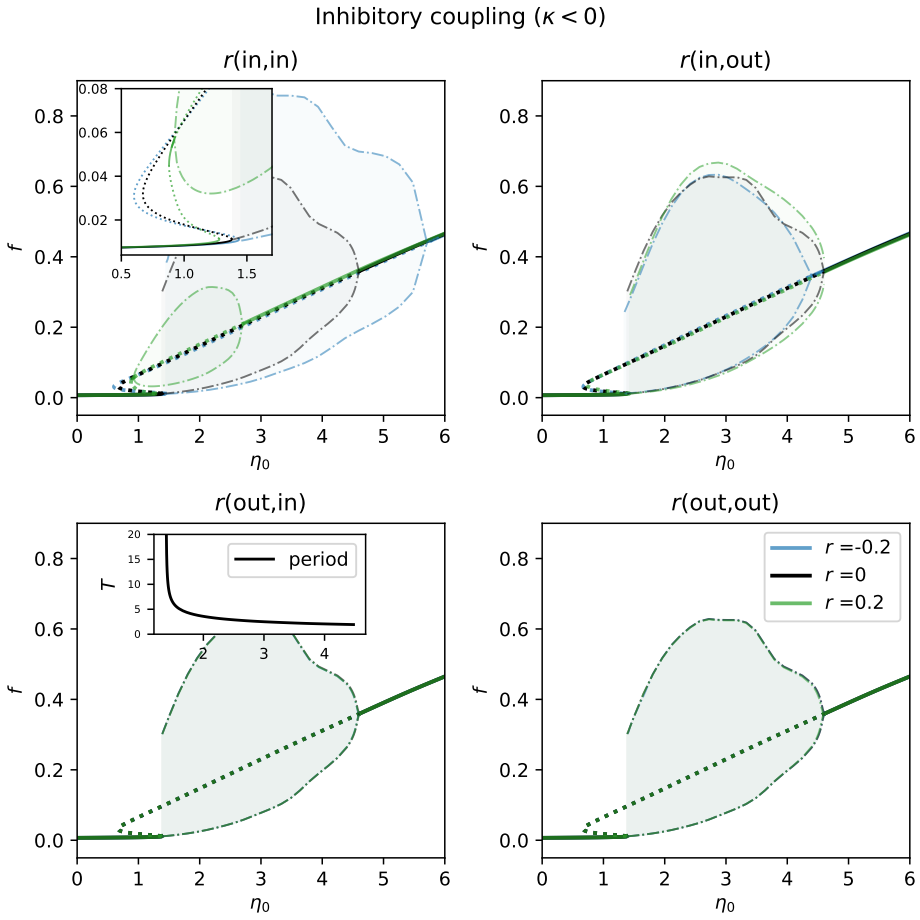

Figure 3.7: Average firing rate at fixed points of (3.2.27)-(3.2.28) as a function of η_0, for the 4 types of assortativity. For each type of assortativity curves are plotted for $r = 0$ (black), $r = -0.2$ (blue) and $r = 0.2$ (green). In addition there are oscillations in certain regions and dash-dotted lines outline the minimal and maximal firing rate over one period of oscillation. The (in,in)-plot in the top left corner contains a zoom of rest of the panel, and the (out,in)-plot contains a subplot with the oscillation's period for $r = 0$ and which is aligned with the outer η_0 axis.

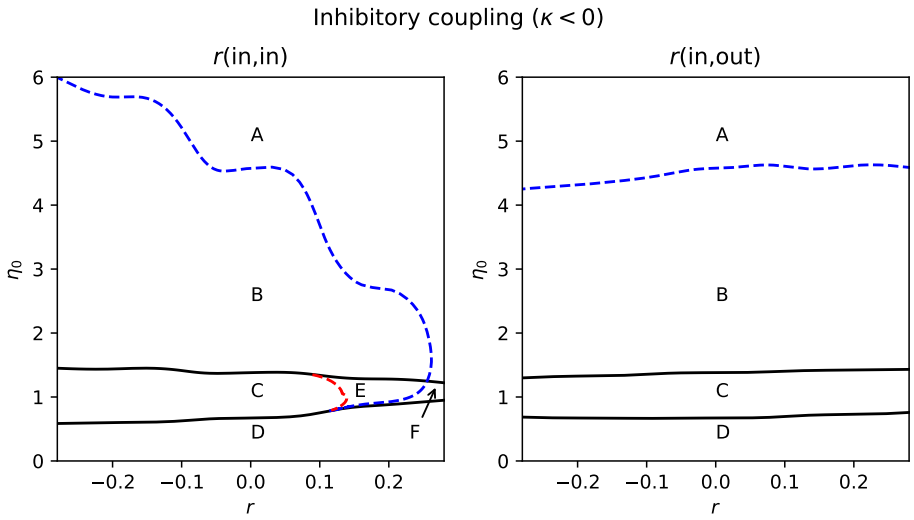

Figure 3.8: Continuation of bifurcations seen in upper panels of Fig. 3.7. Solid black lines indicate saddle-node bifurcations, dashed blue is a Hopf bifurcation and dashed red a homoclinic bifurcation. Curves in Figure 3.7 can be understood as vertical slices through the respective plot at $r = 0, \pm 0.2$. See text for explanation of labels.

neuron's degree and that the η_j are chosen from a Lorentzian, and using the Ott/Antonsen ansatz, they derived equations similar to (3.2.27)-(3.2.28). The difference in formulations is that rather than a sum over entries of E (in (3.2.28)), [CHC+17] wrote the sum as

$$\sum_{\boldsymbol{k}'} P(\boldsymbol{k}')a(\boldsymbol{k}' \to \boldsymbol{k}) \tag{3.6.1}$$

where $P(\boldsymbol{k})$ is the degree distribution and $a(\boldsymbol{k}' \to \boldsymbol{k})$ is the assortativity function, which specifies the probability of a link from a node with degree $\boldsymbol{k}'$ to one with degree $\boldsymbol{k}$ (given that such neurons exist). They then chose a particular functional form for a and briefly presented the results of varying one type of assortativity (between k'_{in} and k_{out}). In contrast, our approach is far more general (since *any* connectivity matrix A can be reduced to the corresponding E, the only assumption being that the dynamics are determined by a neuron's degree). We also show the results of a wider investigation into the effects of assortativity.

This alternative presentation also explains why E can be well approximated with a low-rank approximation. If the in- and out-degrees of a single neuron are independent, $P(\boldsymbol{k}') = P_i(k'_{in})P_o(k'_{out})$, and with neutral assortativity, $a(\boldsymbol{k}' \to \boldsymbol{k}) = k'_{out}k_{in}/(N\langle k\rangle)$. Thus

$$\sum_{\boldsymbol{k}'} P(\boldsymbol{k}')a(\boldsymbol{k}' \to \boldsymbol{k})H(b(\boldsymbol{k}'); q) = \frac{k_{in}}{N\langle k\rangle} \sum_{k'_{out}} \sum_{k'_{in}} k'_{out}P_i(k'_{in})P_o(k'_{out})H(b(k'_{out}, k'_{in}); q) \tag{3.6.2}$$

This term contributes to the input current to a neuron with degree $\boldsymbol{k} = (k_{in}, k_{out})$, but is independent of k_{out}. Thus the state of a neuron depends only on its in-degree, so

$$\sum_{\boldsymbol{k}'} P(\boldsymbol{k}')a(\boldsymbol{k}' \to \boldsymbol{k})H(b(\boldsymbol{k}');q) = \frac{k_{in}}{N}\sum_{k'_{in}} P_i(k'_{in})H(b(k'_{in});q) \tag{3.6.3}$$

Comparing with (3.2.28) we see that $E = \mathbf{c}^T\mathbf{d}$ where $\mathbf{c} = (k_{in}^1, k_{in}^2 \dots k_{in}^{N_{k^{in}}})/N$ and $\mathbf{d} = (P_i(k_{in}^1), P_i(k_{in}^2) \dots, P_i(k_{in}^{N_{k^{in}}}))$, that is E is a rank-one matrix. Varying assortativity within the network is then a perturbation away from this, with the effects appearing in the second (and third) singular values in the SVD decomposition of E.

A limitation of our study is that we considered only networks of fixed size with the same distributions of in- and out-degrees, and a specific distribution of these degrees. However, our approach does not rely on this and could easily be adapted to consider other types of networks, although we expect it to become less valid as both the average degree and number of neurons in the network decrease. We have also only considered theta neurons, but since a theta neuron is the normal form of a type I neuron, we expect similar networks of other type I neurons to behave similarly to the networks considered here. The approach presented here could also be used to efficiently investigate the effects of correlated heterogeneity, where either the mean or width of the distribution of the η_j is correlated with a neuron's in- or out-degree [SSTR13, SSSG13, CGDM13]. We could also consider assortativity based on a neuron's intrinsic drive (η_j) [SRO15] rather than its degrees, or correlations between an individual neuron's in- and out-degree [VHT13, LS10, MHT17, VR19, NFS$^+$17].

Chapter 4

Numerical methods

4.1 Singular value decomposition

Studying large or detailed systems often involves big matrices. Performing algebraic operations with them can become computationally costly and thus time consuming. Especially in cases where matrix entries show an inherent relation to one another, it can be feasible to approximate such a matrix. The singular value decomposition (SVD)[GR71] refers to a factorization of a matrix $E \in \mathbb{R}^N \times \mathbb{R}^N$ into three factors

$$E = USV^T. \tag{4.1.1}$$

The matrices U and V are orthonormal. They consist of sets of N orthonormal basis functions u_i and v_i with $u_i \cdot u_j = v_i \cdot v_j = \delta_{ij}$. With S we denote a diagonal matrix with decreasing entries, called singular values. There are several ways of computing such a decomposition [Wat91]. Having this particular shape, the matrix E can be expressed as

$$E = \sum_{i=1}^{N} s_i(u_i \cdot v_i). \tag{4.1.2}$$

As u_i and v_i are normalised, the relevance of the ith term can be estimated by the size of s_i. If the singular values decrease quickly enough, one can approximate the sum in Equation 4.1.2 by neglecting all terms but the M first, which are the M most relevant terms:

$$E \approx \sum_{i=1}^{M} s_i(u_i \cdot v_i). \tag{4.1.3}$$

The approximated matrix will show key features of the original one as depicted in Figure 4.1.

Figure 4.1: Demonstration of approximating a matrix using SVD. The original matrix storing grey values of a 400x400 pixel image is illustrated in the bottom right corner. Using as little as $M = 2$ basis functions only very basic features are captured in the approximation. One can observe how details increase with larger values of M.

Matrix-vector multiplication One advantage of having E in the form of Equation 4.1.3 is the computational efficiency when multiplying E with a vector $p \in \mathbb{R}^N$. In general, $E \cdot p$ will require N^2 multiplications and the same number of additions. Utilising an approximated E we write

$$E \cdot p \approx \sum_{i=1}^{M} s_i (u_i \cdot v_i) \cdot p \tag{4.1.4}$$

$$\approx \sum_{i=1}^{M} s_i u_i \underbrace{(v_i \cdot p)}_{\in \mathbb{R}}, \tag{4.1.5}$$

where $v_i \cdot p$ gets computed first and the result is then multiplied with s_i and subsequently with each component of u_i. This procedure requires only $M(N + 2)$ multiplications and

MN additions. In section 3, we use $N = 100$ and $M = 3$, and therefore the approximated multiplication requires almost two orders of magnitude fewer operations.

4.2 Pseudo arc-length continuation

Suppose we study a dynamical system $\frac{d\boldsymbol{b}}{dt} = F(\boldsymbol{b}, x)$, where $\boldsymbol{b}$ is a multi-dimensional vector of variables and x a parameter of the system. Often, steady state solutions (fixed points) and their dependence on system parameters are of interest. If we are unable to solve $F(\boldsymbol{b}, x) = 0$ analytically, numerical time integration can help to find stable solutions. In case of a more complex structured phase space and depending on initial conditions, we may even find several of those solutions for a given set of parameters. However, in general it will not be possible to compute unstable solutions. For a range of parameters x the equation $F(\boldsymbol{b}, x) = 0$ describes a curve of solutions. It is a tedious procedure to run a time integration for each value of x, since the system has to reach equilibrium which may take a lot of time. A clever simplification of this lengthy brute-force procedure is the scheme presented below. It is based on finding a solution close to a known one by stepping along the tangent of the solution curve and employing Newton's method to quickly converge towards a steady state solution. This method is called *Pseudo arc-length continuation*. In addition to its efficiency it allows to find unstable solutions as well. In this section we will have a look at it in detail following [Lai14b] and discuss how it can be used to track bifurcation points when varying a second parameter. The latter will be similar to [SBRP+02], with the difference of using pseudo arc-length instead of zero-order continuation.

4.2.1 Single parameter

Consider a system described by a set of variables $\boldsymbol{b} \in \mathbb{R}^N$. Its dynamics depend on a parameter x and read

$$\frac{d\boldsymbol{b}}{dt} = F(\boldsymbol{b}, x) \tag{4.2.1}$$

such that for all steady state solutions

$$F(\boldsymbol{b}, x) = 0 \tag{4.2.2}$$

Eventually, we are interested in a curve or series of points for which Equation 4.2.2 holds. The scheme starts with a time integration where $x = x_0$ leads us to a stable steady state $\boldsymbol{b}_0$. The point $(\boldsymbol{b}_0, x_0)$ is the first on the curve of interest as depicted in Figure 4.2. Note that the vertical axis represents the space of $\boldsymbol{b}$ which is $\mathbb{R}^N$ where N can be larger than 1.

Next, we find the tangent vector $(\dot{\boldsymbol{b}}_0, \dot{x}_0)$. With $F_{\boldsymbol{b}}$ and F_x being the partial derivatives of F with respect to $\boldsymbol{b}$ and x respectively, and evaluated at $(\boldsymbol{b}_0, x_0)$, we compute the null-vector of the matrix $(F_{\boldsymbol{b}}|F_x)$ using singular value decomposition, where we take the vector

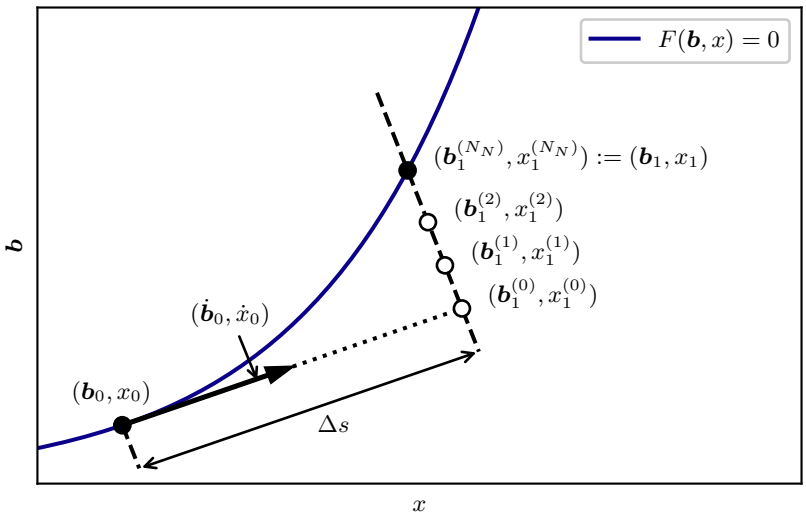

Figure 4.2: Schematic illustration of pseudo arc-length continuation. Starting from an initial point $(\boldsymbol{b}_0, x_0)$ one follows $(\dot{\boldsymbol{b}}_0, \dot{x}_0)$, the tangent of the curve $F(\boldsymbol{b}, x)$, for a chosen length Δs. From that point $(\boldsymbol{b}_1^{(0)}, x_1^{(0)})$ we use Newton's method to find another stationary solution $(\boldsymbol{b}_1, x_1)$ under the constraint that we only go along a line perpendicular to the tangent.

corresponding to the smallest singular value. Note, that this vector has to be normalised to length 1. Subsequently, we make a step of size Δs from $(\boldsymbol{b}_0, x_0)$ along the tangent $(\dot{\boldsymbol{b}}_0, \dot{x}_0)$ to

$$\boldsymbol{b}_1^{(0)} = \boldsymbol{b}_0 + \dot{\boldsymbol{b}}_0 \Delta s \tag{4.2.3}$$

$$x_1^{(0)} = x_0 + \dot{x}_0 \Delta s \tag{4.2.4}$$

This point is quite close to a steady state such that we can use it as starting point to employ Newton's method for a quicker convergence compared to time integration. To guarantee that the scheme indeed converges to a point a bit further down the curve, an additional constraint is applied: we require the solution of each Newton iteration $(\boldsymbol{b}_1^{(i)}, x_1^{(i)})$ to lie on a hyperplane perpendicular to the tangent $(\dot{\boldsymbol{b}}_0, \dot{x}_0)$ with distance Δs, thus

$$\Delta s = \begin{pmatrix} (\boldsymbol{b}_1^{(i)} - \boldsymbol{b}_0) \\ (x_1^{(i)} - x_0) \end{pmatrix} \cdot \begin{pmatrix} \dot{\boldsymbol{b}}_0 \\ \dot{x}_0 \end{pmatrix} \qquad \text{or} \tag{4.2.5}$$

$$0 = (\boldsymbol{b}_1^{(i)} - \boldsymbol{b}_0)^T \dot{\boldsymbol{b}}_0 + (x_1^{(i)} - x_0)^T \dot{x}_0 - \Delta s \tag{4.2.6}$$

where T refers to transpose.

Newton's Method for solving $F(\boldsymbol{b}_1, x_1) = 0$ under the constraint (4.2.6) reads

$$\begin{pmatrix} \boldsymbol{b}_1^{(i+1)} \\ x_1^{(i+1)} \end{pmatrix} = \begin{pmatrix} \boldsymbol{b}_1^{(i)} \\ x_1^{(i)} \end{pmatrix} - J_{(i)}^{-1} \begin{pmatrix} F(\boldsymbol{b}_1^{(i)}, x_1^{(i)}) \\ (\boldsymbol{b}_1^{(i)} - \boldsymbol{b}_0)^T \dot{\boldsymbol{b}}_0 + (x_1^{(i)} - x_0)^T \dot{x}_0 - \Delta s \end{pmatrix} \tag{4.2.7}$$

where

$$J_{(i)} = \begin{pmatrix} F_{\boldsymbol{b}} & F_x \\ \dot{\boldsymbol{b}}_0 & \dot{x}_0 \end{pmatrix} \tag{4.2.8}$$

evaluated at $(\boldsymbol{b}_1^{(i)}, x_1^{(i)})$. The index i runs from 0 to $M-1$, the number of iterations for which we assume the system has converged and we define $(\boldsymbol{b}_1^{(M)}, x_1^{(M)}) := (\boldsymbol{b}_1, x_1)$.

From here we repeat the procedure and gain $(\boldsymbol{b}_2, x_2)$ and so on. The stability of each steady state can be concluded from the eigenvalues of $F_{\boldsymbol{b}}$.

Differentiating the complex conjugate Having said that $\boldsymbol{b} \in \mathbb{R}^N$, we realise that the state vector of an ensemble network or degree mean field is actually complex, since the imaginary unit appears in its dynamical equation. Let $\boldsymbol{\beta}$ be such a state vector with $\boldsymbol{\beta} \in \mathbb{C}^L$ where L is the number of neurons in the case of ensemble equations, or the number of distinct degrees, degree clusters or virtual degrees in a degree mean field system. In general, complex dynamical equations pose no problem, but the mean field dynamics of a theta neuron network include the complex conjugate of state variables, which is a non-analytical function and thus not complex differentiable. A solution is provided by separating the set of complex variables in twice as many real states $\boldsymbol{b} \in \mathbb{R}^{2L}$, where $2L = N$. Let

$$\boldsymbol{b}^{\mathrm{Re}} = \mathrm{Re}\,(\boldsymbol{\beta}) \qquad \text{and} \qquad \boldsymbol{b}^{\mathrm{Im}} = \mathrm{Im}\,(\boldsymbol{\beta}) \tag{4.2.9}$$

such that

$$\boldsymbol{b} = \begin{pmatrix} \boldsymbol{b}^{\mathrm{Re}} \\ \boldsymbol{b}^{\mathrm{Im}} \end{pmatrix} \tag{4.2.10}$$

and

$$\frac{\partial \boldsymbol{b}}{\partial t} = \begin{pmatrix} \mathrm{Re}\left(\frac{\partial \boldsymbol{\beta}}{\partial t}\right) \\ \mathrm{Im}\left(\frac{\partial \boldsymbol{\beta}}{\partial t}\right) \end{pmatrix} \tag{4.2.11}$$

This workaround allows one to use the pseudo arc-length continuation scheme as described above.

Null vector direction Gaining the tangent (or null vector) from Singular Value Decomposition it may not point in the desirable direction, but the opposite. Computing the first tangent $(\dot{\boldsymbol{b}}_0, \dot{x}_0)$, we simply make sure that its last component $\dot{x}_0$ is positive (negative) if we

would like to trace the continuation curve in positive (negative) x direction.
Throughout the scheme, we might face a null vector pointing just backwards, subsequently leading to either going back along the curve or stepping back and forth at that point in space. Consider step i on the curve. To avoid this behaviour we compute the scalar product of the current null vector $(\dot{\boldsymbol{b}}_i, \dot{x}_i)$ and the previous one $(\dot{\boldsymbol{b}}_{i-1}, \dot{x}_{i-1})$. If the product is < 0 we multiply $(\dot{\boldsymbol{b}}_i, \dot{x}_i)$ by -1:

$$(\dot{\boldsymbol{b}}_i, \dot{x}_i) \cdot (\dot{\boldsymbol{b}}_{i-1}, \dot{x}_{i-1}) \begin{cases} \geqslant 0 : \text{ use } (\dot{\boldsymbol{b}}_i, \dot{x}_i) \\ < 0 : \text{ use } -(\dot{\boldsymbol{b}}_i, \dot{x}_i) \end{cases} \tag{4.2.12}$$

4.2.2 Saddle-node bifurcation

With an extended set of equations we can use pseudo arc-length continuation to track saddle-node bifurcations as a function of a second parameter y. At a saddle-node there is a single eigenvalue $\lambda = 0$ with associated eigenvector $\boldsymbol{n}$. Thus we characterise this bifurcation by

$$F = 0 \tag{4.2.13}$$

$$F_{\boldsymbol{b}}\boldsymbol{n} = 0 \tag{4.2.14}$$

$$\boldsymbol{n}^2 - 1 = 0 \tag{4.2.15}$$

where F and $F_{\boldsymbol{b}}$ are evaluated at $(\boldsymbol{b}, x, y)$. Equation 4.2.13 is the necessary condition for a stationary solution. Equation 4.2.14 gives the null-vector $\boldsymbol{n}$ of the dynamical matrix $F_{\boldsymbol{b}}$, or in other words the eigenvector corresponding to a zero eigenvalue. To ensure we exclude the trivial solution $\boldsymbol{n} = 0$, $\boldsymbol{n}$ has to have an arbitrary non-zero length, for example 1 like in Equation 4.2.15. It has been proven to be advantageous to include $\boldsymbol{n}$ in the set of variables to be continued by the scheme, rather than to repeatedly compute it from $F_{\boldsymbol{b}}$. Therefore our variables are $(\boldsymbol{b}, \boldsymbol{n}, x, y) \in \mathbb{R}^{2N+2}$.

Suppose we have found a saddle node bifurcation at $(\boldsymbol{b}_0, x_0, y_0)$. We compute the null-vector of $F_{\boldsymbol{b}}(\boldsymbol{b}_0, x_0, y_0)$ to gain the full set of initial variables $(\boldsymbol{b}_0, \boldsymbol{n}_0, x_0, y_0)$. In principle, we proceed exactly as above with the difference of an extended set of variables and equations. The first derivatives of Equation 4.2.13-4.2.15 with respect to all independent variables and parameters read

$$\begin{pmatrix} F_{\boldsymbol{b}} & F_{\boldsymbol{n}} & F_x & F_y \\ \frac{\partial F_{\boldsymbol{b}}\boldsymbol{n}}{\partial \boldsymbol{b}} & \frac{\partial F_{\boldsymbol{b}}\boldsymbol{n}}{\partial \boldsymbol{n}} & \frac{\partial F_{\boldsymbol{b}}\boldsymbol{n}}{\partial x} & \frac{\partial F_{\boldsymbol{b}}\boldsymbol{n}}{\partial y} \\ \frac{\partial \boldsymbol{n}^2-1}{\partial \boldsymbol{b}} & \frac{\partial \boldsymbol{n}^2-1}{\partial \boldsymbol{n}} & \frac{\partial \boldsymbol{n}^2-1}{\partial x} & \frac{\partial \boldsymbol{n}^2-1}{\partial y} \end{pmatrix} = \begin{pmatrix} F_{\boldsymbol{b}} & 0 & F_x & F_y \\ \frac{\partial F_{\boldsymbol{b}}\boldsymbol{n}}{\partial \boldsymbol{b}} & F_{\boldsymbol{b}} & \frac{\partial F_{\boldsymbol{b}}\boldsymbol{n}}{\partial x} & \frac{\partial F_{\boldsymbol{b}}\boldsymbol{n}}{\partial y} \\ 0 & 2\boldsymbol{n} & 0 & 0 \end{pmatrix} \tag{4.2.16}$$

and by computing the null-vector of this matrix we gain the tangent $(\dot{\boldsymbol{b}}_0, \dot{\boldsymbol{n}}_0, \dot{x}_0, \dot{y}_0)$. The

initial values for Newton's method are

$$\boldsymbol{b}_1^{(0)} = \boldsymbol{b}_0 + \dot{\boldsymbol{b}}_0 \Delta s \tag{4.2.17}$$

$$\boldsymbol{n}_1^{(0)} = \boldsymbol{n}_0 + \dot{\boldsymbol{n}}_0 \Delta s \tag{4.2.18}$$

$$x_1^{(0)} = x_0 + \dot{x}_0 \Delta s \tag{4.2.19}$$

$$y_1^{(0)} = y_0 + \dot{y}_0 \Delta s \tag{4.2.20}$$

Now, we solve simultaneously Equation 4.2.13-4.2.15 and the perpendicular hyperplane constraint

$$0 = (\boldsymbol{b}_1^{(i)} - \boldsymbol{b}_0)^T \dot{\boldsymbol{b}}_0 + (\boldsymbol{n}_1^{(i)} - \boldsymbol{n}_0)^T \dot{\boldsymbol{n}}_0 + (x_1^{(i)} - x_0)^T \dot{x}_0 + (y_1^{(i)} - y_0)^T \dot{y}_0 - \Delta s \tag{4.2.21}$$

$$= g_1^{(i)} \tag{4.2.22}$$

by computing several iterations of

$$\begin{pmatrix} \boldsymbol{b}_1^{(i+1)} \\ \boldsymbol{n}_1^{(i+1)} \\ x_1^{(i+1)} \\ y_1^{(i+1)} \end{pmatrix} = \begin{pmatrix} \boldsymbol{b}_1^{(i)} \\ \boldsymbol{n}_1^{(i)} \\ x_1^{(i)} \\ y_1^{(i)} \end{pmatrix} - J_{(i)}^{-1} \begin{pmatrix} F \\ F_{\boldsymbol{b}} \boldsymbol{n}_1^{(i)} \\ \left(\boldsymbol{n}_1^{(i)}\right)^2 - 1 \\ g_1^{(i)} \end{pmatrix}, \tag{4.2.23}$$

where

$$J_{(i)} = \begin{pmatrix} F_{\boldsymbol{b}} & 0 & F_x & F_y \\ \frac{\partial F_{\boldsymbol{b}} \boldsymbol{n}_1^{(i)}}{\partial \boldsymbol{b}} & F_{\boldsymbol{b}} & \frac{\partial F_{\boldsymbol{b}} \boldsymbol{n}_1^{(i)}}{\partial x} & \frac{\partial F_{\boldsymbol{b}} \boldsymbol{n}_1^{(i)}}{\partial y} \\ 0 & 2\boldsymbol{n}_1^{(i)} & 0 & 0 \\ \dot{\boldsymbol{b}}_0 & \dot{\boldsymbol{n}}_0 & \dot{x}_0 & \dot{y}_0 \end{pmatrix} \tag{4.2.24}$$

with $F, F_{\boldsymbol{b}}, F_x$, and F_y being evaluated at $(\boldsymbol{b}_1^{(i)}, x_1^{(i)}, y_1^{(i)})$.

Once the scheme is converged to $(\boldsymbol{b}_1, \boldsymbol{n}_1, x_1, y_1)$ we repeat the whole process to find $(\boldsymbol{b}_2, \boldsymbol{n}_2, x_2, y_2)$ and so on. Note that $g_1^{(i)}$ is specific to step 1 and has to be updated for each step.

4.2.3 Hopf bifurcation

At a Hopf bifurcation, a pair of complex conjugate eigenvalues crosses the imaginary axis and has zero real part $\lambda_{1/2} = \pm i\omega$ with associated eigenvector $\boldsymbol{c} \pm i\boldsymbol{d}$. They can be sufficiently

described by the following equations:

$$F = 0 \tag{4.2.25}$$
$$F_{\boldsymbol{b}}\boldsymbol{c} + \omega\boldsymbol{d} = 0 \tag{4.2.26}$$
$$F_{\boldsymbol{b}}\boldsymbol{d} - \omega\boldsymbol{c} = 0 \tag{4.2.27}$$
$$\boldsymbol{\phi}\cdot\boldsymbol{c} - 1 = 0 \tag{4.2.28}$$
$$\boldsymbol{\phi}\cdot\boldsymbol{d} = 0 \tag{4.2.29}$$

with F and $F_{\boldsymbol{b}}$ being evaluated at $(\boldsymbol{b}, x, y)$ and where $\boldsymbol{\phi}$ is a constant vector. The first equation specifies an equilibrium point in general. The second and third equation state that the eigenvalue has no real part, whereas the fourth and fifth equation lock phase and amplitude of the eigenvector. In total the system consists of $3N + 2$ equations and the solution vector is $(\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}, \omega, x, y) \in \mathbb{R}^{3N+3}$. Note that for $\omega = 0$, two of the above equations become redundant and the system is equivalent to the previous ones where we followed a saddle-node bifurcation.

Assume an initial point of bifurcation at $(\boldsymbol{b}_0, x_0, y_0)$. We compute the eigenvalues of $F_{\boldsymbol{b}}(\boldsymbol{b}_0, x_0, y_0)$ and find ω_0 and its corresponding eigenvector $c_0 + id_0$. The constant vector $\boldsymbol{\phi}$ is chosen to be $\boldsymbol{c}_0$, which means it does not change as we determine new points along the curve. Analogous to Equation 4.2.16 we build a matrix of derivatives of the system (4.2.25)-(4.2.29)

$$\begin{pmatrix} F_{\boldsymbol{b}} & 0 & 0 & 0 & F_x & F_y \\ \frac{\partial F_{\boldsymbol{b}}\boldsymbol{c}}{\partial \boldsymbol{b}} & F_{\boldsymbol{b}} & \omega & \boldsymbol{d} & \frac{\partial F_{\boldsymbol{b}}\boldsymbol{c}}{\partial x} & \frac{\partial F_{\boldsymbol{b}}\boldsymbol{c}}{\partial y} \\ \frac{\partial F_{\boldsymbol{b}}\boldsymbol{d}}{\partial \boldsymbol{b}} & -\omega & F_{\boldsymbol{b}} & -\boldsymbol{c} & \frac{\partial F_{\boldsymbol{b}}\boldsymbol{d}}{\partial x} & \frac{\partial F_{\boldsymbol{b}}\boldsymbol{d}}{\partial y} \\ 0 & \boldsymbol{\phi} & 0 & 0 & 0 & 0 \\ 0 & 0 & \boldsymbol{\phi} & 0 & 0 & 0 \end{pmatrix} \tag{4.2.30}$$

and calculate its normalised null-vector to get the tangent $(\dot{\boldsymbol{b}}_0, \dot{\boldsymbol{c}}_0, \dot{\boldsymbol{d}}_0, \dot{\omega}_0, \dot{x}_0, \dot{y}_0)$. Note that the entry ω is actually $\omega \cdot \boldsymbol{I}_N$, with $\boldsymbol{I}_N$ being the $N \times N$ identity matrix. A step of size Δs along this tangent leads us to

$$\boldsymbol{b}_1^{(0)} = \boldsymbol{b}_0 + \dot{\boldsymbol{b}}_0\Delta s \tag{4.2.31}$$
$$\boldsymbol{c}_1^{(0)} = \boldsymbol{c}_0 + \dot{\boldsymbol{c}}_0\Delta s \tag{4.2.32}$$
$$\boldsymbol{d}_1^{(0)} = \boldsymbol{d}_0 + \dot{\boldsymbol{d}}_0\Delta s \tag{4.2.33}$$
$$\omega_1^{(0)} = \omega_0 + \dot{\omega}_0\Delta s \tag{4.2.34}$$
$$x_1^{(0)} = x_0 + \dot{x}_0\Delta s \tag{4.2.35}$$
$$y_1^{(0)} = y_0 + \dot{y}_0\Delta s \tag{4.2.36}$$

from where we apply several iterations of Newton's method. We not only seek to solve equations (4.2.25)-(4.2.29), but also add a constraint to ensure the solution is on the hyperplane perpendicular to the tangent as done in the previous sections 4.2.1 and 4.2.2, which reads in this case

$$\begin{aligned} 0 &= (\boldsymbol{b}_1^{(i)} - \boldsymbol{b}_0)^T \dot{\boldsymbol{b}}_0 + (\boldsymbol{c}_1^{(i)} - \boldsymbol{c}_0)^T \dot{\boldsymbol{c}}_0 + (\boldsymbol{d}_1^{(i)} - \boldsymbol{d}_0)^T \dot{\boldsymbol{d}}_0 + (\omega_1^{(i)} - \omega_0)^T \dot{\omega}_0 \\ &\quad + (x_1^{(i)} - x_0)^T \dot{x}_0 + (y_1^{(i)} - y_0)^T \dot{y}_0 - \Delta s \\ &= g_1^{(i)} \end{aligned} \tag{4.2.37}$$

Newton's method can then be formulated as

$$\begin{pmatrix} \boldsymbol{b}_1^{(i+1)} \\ \boldsymbol{c}_1^{(i+1)} \\ \boldsymbol{d}_1^{(i+1)} \\ \omega_1^{(i+1)} \\ x_1^{(i+1)} \\ y_1^{(i+1)} \end{pmatrix} = \begin{pmatrix} \boldsymbol{b}_1^{(i)} \\ \boldsymbol{c}_1^{(i)} \\ \boldsymbol{d}_1^{(i)} \\ \omega_1^{(i)} \\ x_1^{(i)} \\ y_1^{(i)} \end{pmatrix} + J_{(i)}^{-1} \begin{pmatrix} F \\ F_{\boldsymbol{b}} \boldsymbol{c}_1^{(i)} + \omega \boldsymbol{d}_1^{(i)} \\ F_{\boldsymbol{b}} \boldsymbol{d}_1^{(i)} - \omega \boldsymbol{c}_1^{(i)} \\ \boldsymbol{\phi} \cdot \boldsymbol{c}_1^{(i)} - 1 \\ \boldsymbol{\phi} \cdot \boldsymbol{d}_1^{(i)} - 1 \\ g_1^{(i)} \end{pmatrix} \tag{4.2.38}$$

where

$$J_{(i)} = \begin{pmatrix} F_{\boldsymbol{b}} & 0 & 0 & 0 & F_x & F_y \\ \frac{\partial F_{\boldsymbol{b}} \boldsymbol{c}_1^{(i)}}{\partial \boldsymbol{b}} & F_{\boldsymbol{b}} & \omega & \boldsymbol{d}_1^{(i)} & \frac{\partial F_{\boldsymbol{b}} \boldsymbol{c}_1^{(i)}}{\partial x} & \frac{\partial F_{\boldsymbol{b}} \boldsymbol{c}_1^{(i)}}{\partial y} \\ \frac{\partial F_{\boldsymbol{b}} \boldsymbol{d}_1^{(i)}}{\partial \boldsymbol{b}} & -\omega & F_{\boldsymbol{b}} & -\boldsymbol{c}_1^{(i)} & \frac{\partial F_{\boldsymbol{b}} \boldsymbol{d}_1^{(i)}}{\partial x} & \frac{\partial F_{\boldsymbol{b}} \boldsymbol{d}_1^{(i)}}{\partial y} \\ 0 & \boldsymbol{\phi} & 0 & 0 & 0 & 0 \\ 0 & 0 & \boldsymbol{\phi} & 0 & 0 & 0 \\ \dot{\boldsymbol{b}}_0 & \dot{\boldsymbol{c}}_0 & \dot{\boldsymbol{d}}_0 & \dot{\omega}_0 & \dot{x}_0 & \dot{y}_0 \end{pmatrix} \tag{4.2.39}$$

with $F, F_{\boldsymbol{b}}, F_x$, and F_y being evaluated at $(\boldsymbol{b}_1^{(i)}, x_1^{(i)}, y_1^{(i)})$.

Once converged to $(\boldsymbol{b}_1, \boldsymbol{c}_1, \boldsymbol{d}_1, \omega_1, x_1, y_1)$, one can progress with the next step, keeping in mind that $g_1^{(i)}$ needs to be altered every step and then updated at each iteration.

4.3 Gaussian quadrature

Well known as an approximation for integrals, n-point Gaussian quadrature can also be utilised to approximate a large sum over N terms with a sum over many fewer, n, terms

$$\sum_{i=1}^{N} f(k_i) \approx \sum_{j=1}^{n} w_j f(\tilde{k}_j) \qquad \text{with } N \gg n \tag{4.3.1}$$

where we evaluate the same function at a new set of arguments $\tilde{k}_j$ and weight those values with the respective w_j.

Consider the set orthogonal polynomials

$$\langle q_\mu(k), q_\nu(k)\rangle = 0 \text{ if } \mu \neq \nu \tag{4.3.2}$$

with the scalar product

$$\langle q_\mu(k), q_\nu(k)\rangle = \sum_{i=1}^{N} q_\mu(k_i) q_\nu(k_i) \tag{4.3.3}$$

It can be shown that if the n nodes $\tilde{k}_j$ are chosen to be the roots of $q_n(k)$, then there exist weights w_j such that the approximation is exact for all polynomials up to degree $2n-1$.

There are several methods to determine such a set of polynomials. We make use of the three-term recurrence relation as it provides an efficient way to compute the weights as well.

Three-Term Recurrence Relation Polynomials get iteratively constructed as follows

$$q_{\mu+1}(k) = (k - \alpha_\mu) q_\mu(k) - \beta_\mu q_{\mu-1}(k) \tag{4.3.4}$$

where

$$\alpha_\mu = \frac{\langle k \cdot q_\mu(k), q_\mu(k)\rangle}{\langle q_\mu(k), q_\mu(k)\rangle} \tag{4.3.5}$$

$$\beta_\mu = \frac{\langle q_\mu(k), q_\mu(k)\rangle}{\langle q_{\mu-1}(k), q_{\mu-1}(k)\rangle} \tag{4.3.6}$$

beginning with

$$q_{-1}(k) = 0 \text{ and } q_0(k) = 1 \tag{4.3.7}$$

Among the several options to compute the nodes $\tilde{k}_j$ and weights w_j from those n polynomials the Golub-Welsch Algorithm is the most popular one.

Golub-Welsch Algorithm The authors of [GW69] form the matrix

$$J = \begin{pmatrix} \alpha_0 & \sqrt{\beta_1} & 0 & 0 & \dots & 0 \\ \sqrt{\beta_1} & \alpha_1 & \sqrt{\beta_2} & 0 & \dots & 0 \\ 0 & \sqrt{\beta_2} & \alpha_2 & \sqrt{\beta_3} & \dots & 0 \\ \vdots & & & & & \vdots \\ 0 & \dots & \sqrt{\beta_{n-3}} & \alpha_{n-3} & \sqrt{\beta_{n-2}} & 0 \\ 0 & \dots & 0 & \sqrt{\beta_{n-2}} & \alpha_{n-2} & \sqrt{\beta_{n-1}} \\ 0 & \dots & 0 & 0 & \sqrt{\beta_{n-1}} & \alpha_{n-1} \end{pmatrix} \tag{4.3.8}$$

and consider the eigenvalues and normalised eigenvectors of J, that is $J\phi_j = \lambda_j \phi_j$. The eigenvalues λ_j turn out to be the roots of $q_n(k)$ and are consequently $\tilde{k}_j$. The weights are computed by

$$w_j = (k_N - k_1) \left(\phi_j^{(1)}\right)^2 \tag{4.3.9}$$

where $\phi_j^{(1)}$ is the first component of the eigenvector.

Chapter 5

Python3 module: *ThetaNet*

All results for this thesis have been computed using Python3 scripts. The core functionality is available as an open source module called *ThetaNet* on github: `https://github.com/cblasche/ThetaNet`.
The *ThetaNet* module comprises the following features:

- Create correlated (Gaussian copula) and uncorrelated degree sequences
- Create adjacency matrices, remove self- and multi-edges, manipulate degree assortativity
- Integrate networks of theta neurons
- Integrate degree mean field dynamics of theta neuron networks
- Numerically continue solutions with respect to coupling strength, intrinsic excitability, and degree correlation/assortativity

It is licenced under the GNU General Public License v2.0.

Installation

If you do not have Python3 installed yet, get it from `https://www.python.org/downloads/`.

- Install the package *setuptools* from `https://pypi.org/project/setuptools/` or if you have the *pip* installed run:

  ```
  pip3 install setuptools
  ```

- Download the repository from `https://github.com/cblasche/thetanet` or if you have *git* installed run:

```
git clone https://github.com/cblasche/thetanet
```

- In the same folder as above run:

```
python3 setup.py install
```

Structure

- **thetanet.dynamics** contains differential equations and integrator setup for a network of Theta neurons, the ensemble equations and the mean field equations.
- **thetanet.generate** holds functions to generate degree sequences, adjacency matrices and assortativity functions. Additionally, there are functions to compute degree correlations and assortativity coefficients.
- **thetanet.continuation** stores various continuation schemes.
- **thetanet.utils** has functions around the topics: correlated probability function, polynomial chaos expansion, and singular value decomposition.

Parameter file

Due to the larger number of parameters in functions related to the dynamics, that is time integration or continuation, we require a parameter file. For convenience, the file is chosen to be a python file, for example the file found in the repository "example/parameters.py". If stored in the same folder as the main script it is imported and renamed by

```
import parameters as pm
```

We can now pass all parameters at once to a function and access single parameters using the "dot"-operator.

```
def f(pm):
    print("Number of neurons:", pm.N)
```

This example illustrates that variable names in the parameter file have to match the specific ones used in the ThetaNet module. Those names can be found when inspecting the source code or in the example parameter file. Often it is feasible to alter parameters within your execution script, which can be done simply by:

```
pm.N = 1000
f(pm)
# >>> Number of neurons: 1000

pm.N = 2000
f(pm)
# >>> Number of neurons: 2000
```

5.1 thetanet.generate

Sequences Suppose we would like to create degree squences for N neurons from a given degree space and distribution with a certain degree correlation ρ. We write

```
# modules
import numpy as np
import thetanet as tn

# degree space
k_in = np.arange(200, 400)
k_out = np.copy(k_in)

# marginal power law degree distribution
P_k_in = k_in.astype(float)**(-3)
P_k_in /= P_k_in.sum()
P_k_out = np.copy(P_k_in)

# correlated degree distribution
rho = 0.4
P_k_func = tn.utils.bivar_pdf_rho(k_in, P_k_in, k_out, P_k_out)
P_k = P_k_func(rho)

# degree sequences
N = 1000
K_in, K_out = tn.generate.degree_sequence_copula(P_k, N, k_in, k_out)
```

Configuration model Having the sequences one can generate adjacency matrices employing the configuration model. Consider a simple network with no self- and multi-edges and (in,out) assortativity. The convention is that an edge A_{ij} is directed from neuron j to neuron i. Degree assortativity of type (in,out) means correlation between the properties in-degree of the sending neuron (j) and out-degree of the receiving neuron (i); hence the ThetaNet naming `j_prop` and `i_prop`.

```
# for a simple network
A = tn.generate.configuration_model(K_in, K_out)

# for a network containing self- and multi-edges
A = tn.generate.configuration_model(K_in, K_out, simple=False)

# for a simple assortative network
r = 0.3
j_prop = 'in'
i_prop = 'out'
A = tn.generate.configuration_model(K_in, K_out, r, i_prop, j_prop)
```

Internal operations can also be called explicitly on an existing matrix A.

```
tn.generate.remove_self_edges(A)
tn.generate.remove_multi_edges(A)
tn.generate.assortative_mixing(A, r, i_prop, j_prop)

# assortative mixing will create multi-edges, which can also be kept
tn.generate.assortative_mixing(A, r, i_prop, j_prop, eliminate_multi_edges=
    False)
```

Chung-Lu model We can also employ the Chung-Lu model. To achieve a non-vanishing assortativity, we either perturb the target connection probabilities with a parameter c or create a neutral network and apply the mixing algorithm.

```
# perturb connectivity probability
c = 2
A = tn.generate.chung_lu_model(K_in, K_out, c, i_prop, j_prop)

# apply mixing algorithm
A = tn.generate.chung_lu_model(K_in, K_out)
tn.generate.assortative_mixing(A, r, i_prop, j_prop)
```

Measures There are functions implemented to compute the degree correlation ρ and the assortativity coefficient r from matrices and sequences.

```
# degree correlation rho
rho = tn.generate.rho_from_sequences(K_in, K_out)
rho = tn.generate.rho_from_matrix(A)

# assortativity coefficient r
r = tn.generate.r_from_matrix(A, i_prop, j_prop)
```

Assortativity function ThetaNet supports three different approaches when computing degree mean field equations where each requires a different assortativity function.

1. The assortativity function of the authors of [CHC+17] depends on the assortativity parameter c and utilizes the full degree space.

   ```
   a = tn.generate.a_func_linear(k_in, k_out, P_k, N, c, i_prop, j_prop)
   # or without assortativity
   a = tn.generate.a_func_linear(k_in, k_out, P_k, N)
   ```

2. In case of large degree spaces it may be useful to coarse-grain the degree space first and simulate a network of virtual degrees of size $N_{\mu_{\text{in}}} \times N_{\mu_{\text{out}}}$. This approach additionally requires the degree probability of those virtual degrees as well as weights to accurately approximate sums over the degree space.

```
# virtual degrees and their weights
N_mu_in = 10
k_v_in, w_in, q_in = tn.utils.three_term_recursion(N_mu_in, k_in, P_k_in)
k_v_out, w_out = np.copy((k_v_in, w_in))
w = np.outer(w_in, w_out)

# virtual degree distribution
import scipy.interpolate as si # required for 2d interpolation
P_k_v_func = lambda rho: si.interp2d(k_in, k_out, P_k_func(rho))(k_v_in,
    k_v_out)
P_k_v = P_k_v_func(rho)

# virtual degree space: assortativity function
a_v = tn.generate.a_func_linear(k_v_in, k_v_out, w*P_k_v, N, c, i_prop,
    j_prop)
```

3. The third approach is used in [BML20], where the adjacency matrix A is transformed by matrix multiplication into $E = CAB$. Here, introducing degree clusters can be a favourable option when dealing with large degree spaces. The number of clusters are $N_{c_{\text{in}}}$ and $N_{c_{\text{out}}}$. There are two implemented ways to create the binning of the degree space, either "linear" or using the cumulated sum of the degree probability ("cumsum").

```
N_c_in, N_c_out = 10, 10
E, B, c_in, c_out = tn.generate.a_func_transform(A, N_c_in, N_c_out,
    mapping='cumsum')
```

The variables `c_in` and `c_out` are representative degrees of the respective cluster.

5.2 thetanet.dynamics

Full neuronal model Suppose we want to integrate the full Theta neuron model, where each neuron gets assigned a node in the network. The parameter file requires the definition of A and N as well as the following variables:

```
# coupling strength
kappa = 3

# pulse function
n = 2  # sharpness parameter
d_n = 2 ** n * (np.math.factorial(n)) ** 2 \
      / float(np.math.factorial(2 * n))  # normalisation factor
Gamma = tn.dynamics.degree_network.Gamma(n)  # ensemble coefficients

# eta's drawn from Lorentzian (Cauchy) distribution
from scipy.stats import cauchy
```

```
eta_0 = -2  # centre of distribution
delta = 0.1  # width of distribution
eta = cauchy.rvs(eta_0, delta, size=N)

# time
t = np.linspace(0, 30, 1000)
```

After having defined all parameters in parameters.py we simply call:

```
import thetanet as tn
import paramters as pm

# start from uniformly distributed angles
theta_t = tn.dynamics.node_network(pm)  # axis 0: time, axis 1: neurons

# start from last time step
theta_t = tn.dynamics.node_network(pm, init=theta_t[-1])

# integrate ensemble equations
z_t = tn.dynamics.node_network_mean(pm)
```

Degree mean field The degree mean field can be computed utilizing one of three approaches related to the respective assortativity function.

1. Additionally, for a full degree space integration the parameter file has to contain the assortativity function `a` and degree probability `P_k`:

   ```
   import thetanet as tn
   import parameters as pm  # parameter file additionally requires: a, P_k

   pm.degree_approach = 'full'

   # start from zero
   b_t = tn.dynamics.degree_network.integrate(pm)  # axis 0: time, axis 1:
       degrees

   # start from last time step
   b_t = tn.dynamics.node_network(pm, init=b_t[-1])
   ```

   Often, the dynamics will only depend on the in-degree such that it is sufficient to neglect out-degrees. This can be done by choosing

   ```
   degree_approach = 'full_in_only'
   ```

2. The same applies to virtual degrees, except that we require `a_v`, `w` and `P_k_v` in the parameter file and

   ```
   degree_approach = 'virtual'
   # or for only considering in-degrees
   degree_approach = 'virtual_in_only'
   ```

3. For the third approach note that the structure in E can be well approximated by utilizing singular value decomposition, which we want to do in this case. In the parameter file where `E` is defined we add:

```
# for rank 3 approximation
usv = tn.utils.usv_from_E(E)
# for arbitrary rank m
m = 5
usv = tn.utils.usv_from_E(E, m=m)
```

This approach is selected by

```
degree_approach = 'transform'
```

5.3 thetanet.continuation

The continuation submodule allows one to continue steady state solution in various ways using pseudo arc-length continuation. The following methods are only available for the degree mean field dynamics, although ensemble equations of the neuronal network could theoretically be continued. Just like in the previous section for the degree mean field we specify `degree_approach` to be either `virtual` or `transform`. In both cases, we can continue solutions when varying coupling strength κ and parameters of the distribution η_0 and Δ.

The `virtual` approach When choosing ρ or c as continuation variable, a respective function has to be defined in the parameter file

```
# for node correlation rho
w_func = tn.utils.bivar_pdf_rho(k_v_in, w_in, k_v_out, w_out)
# such that w = w_func(rho)

# for assortativity r, or c respectively
a_v_func = lambda c: tn.generate.a_func_linear_r(k_v_in, k_v_out, w, N, c,
    i_prop, j_prop)
# such that a_v = a_v_func(c)
```

The `transform` approach We have seen in the dynamics section, that this mean field approach relies on SVD and we have to provide vectors and singular values u, s and v. Degree correlation and assortativity are implicit variables and thus we interpolate between different matrices with different values of ρ and r. Suppose we have generated a list of adjacency matrices `A_list` according to a list of assortativity values `r_list`, then we can pack the SVD essentials in a file containing all relevant data.

```
import numpy as np
```

```
# generate B, c_in, c_out from an arbitrary adjacency matrix of A_list
B, c_in, c_out = tn.generate.a_func_transform(A_list[0], pm.N_c_in, pm.
    N_c_out)[1:]

# decomposition
e_list = tn.utils.essential_list_from_data(A_list, pm.N_c_in, pm.N_c_out)
np.savez('svd_list', u=e_list[0], s=e_list[1], v=e_list[2], r_list=r_list,
         B=B, c_in=c_in, c_out=c_out)
```

In the parameter file we load the file "svd_list.npz" and create a function, such that `usv` can be computed for an arbitrary `r` within the range of `r_list`.

```
svd_list = np.load('svd_list.npz')
usv_func = tn.utils.usv_func_from_svd_list(svd_list)
# such that usv = usv_func(r)
```

If one is interested in altering ρ the same process applies. There is no separate function implemented to do so, instead `A_list` has to be a list of adjacency matrices with different degree correlation according to `r_list`, which is actually a list of ρ values.

Due to the nature of this approach ρ and any kind of r can only be continued one at a time.

5.3.1 Single-parameter continuation

For the scheme we define necessary functions as stated above and several parameters in the parameter file

```
# variable to be continued can be 'kappa', 'eta_0', 'delta', 'rho' or 'r'
c_var = 'kappa'
c_ds = 0.05  # step size
c_steps = 40  # number of steps
c_n = 7  # number of Newton iterations
```

The Newton method will break as soon as a certain precision is achieved, so `c_n` will only mark the maximum number of iterations. If the scheme never reaches the maximum number of steps it will be break and return the steps done so far. With the internal convention of the parameter being labelled as `x` we compute a continuation curve as follows

```
import parameters as pm
import thetanet as tn

# starting with a time integration from b=0 to a stable fixed point
b_x, x, stable = tn.continuation.single_param(pm)
# b_x; axis 0: curve, axis 1: degrees
# x; continuation variable along the continued curve

# starting at the end of the the first curve
b_x, x, stable = tn.continuation.single_param(pm, init_b=b_x[-1], init_x=x
    [-1], init_stability=stable[-1])
```

Note that by default the scheme uses an adaptive step size, such that the second piece of the curve starts most likely not with step size `c_ds=0.05`. If necessary, it can be reset by calling `pm.c_ds=0.05` in between.

5.3.2 Saddle-node bifurcation tracking

In the event of a saddle-node bifurcation occurring along the continuation curve, we can track it when varying a second parameter, for example r. Therefore we define in the parameter file

```
c_var2 = 'r'  # 2nd continuation variable
```

And in the main script we call

```
import parameters as pm
import thetanet as tn

# computing a continuation curve
b_x, x, stable = tn.continuation.single_param(pm)

# stability change indicates a potential saddle-node bifurcation
sn_list = np.where(stable[:-1] != stable[1:])[0]
# considering only the first one here
sn = sn_list[0]

# reset stepsize
pm.c_ds = 0.05

# internally y is used as label for second parameter
init_b = b_x[sn]
init_x = x[sn]
init_y = pm.r

# continue saddle-nodes
b_xy, x, y = tn.continuation.saddle_node(pm, init_b, init_x, init_y)

# for further continuation of the curve it may be handy to use full states
    including the null vector
bnxy1 = tn.continuation.saddle_node(pm, init_b, init_x, init_y,
    full_state_output=True)
bnxy2 = tn.continuation.saddle_node(pm, init_full_state=bnxy1,
    full_state_output=True)
```

The full continuation state contains the real parts of `b` followed by the imaginary parts of `b`, some internal null vector and the two parameters `x` and `y`. They can be accessed as follows

```
# with N being the number of virtual degrees or degree clusters
b_xy = tn.continuation.complex_unit(bnxy1[:2*N])
x = bnxy1[-2]
```

```
y = bnxy1[-1]
```

5.3.3 Hopf bifurcation tracking

There is another scheme implemented if the change of stability in a continuation curve occurs due to a Hopf bifurcation. We proceed very much in analogy to saddle-node bifurcation tracking, but eventually we call

```
b_xy, x, y = tn.continuation.hopf(pm, init_b, init_x, init_y)
# or using full_output
bcdoxy = tn.continuation.hopf(pm, init_b, init_x, init_y, full_state_output=
    True)
```

When using `full_state_output=True` the function returns a larger vector since this scheme continues an imaginary eigenvalue $\pm$`o` and its complex eigenvector `c`+i`d` instead of the former null vector.

```
# with N being the number of virtual degrees or degree clusters
b_xy = tn.continuation.complex_unit(bcdoxy[:2*N])
x = bcdoxy[-2]
y = bcdoxy[-1]
```

5.4 thetanet.utils

In this submodule, there are auxiliary functions and routines we have already used to some extent.

Three-Term-Recursion The Three-Term-Recursion function takes the maximal polynomial power N_μ as input, as well the space k and the weight function $P(k)$. It returns the N_μ roots of the highest polynomials, the respective weights and a list of polynomials from a constant up to order N_μ.

```
k_v, w, q = tn.utils.three_term_recursion(N_mu, k, P_k)
```

Bivariate probability density function ThetaNet incorporates a Gaussian copula based implementation of correlating marginal probability density functions. Let `pdf_y` and `pdf_z` be two independent probability density functions of the variables `y` and `z`, respectively. Make sure they each sum up to 1. We correlate them by setting the correlation `rho_gauss` to a value between -1 and 1, where we keep in mind that the actual correlation will be slightly different.

```
pdf_yz = tn.utils.bivar_pdf(pdf_y, pdf_z, rho_gauss)
```

```
# getting N samples from it, these will be indices
ind_y, ind_z = sample_from_bivar_pdf(pdf, N)
y_sample = y[ind_y]
z_sample = z[ind_z]

# or similarly use
y_sample, z_sample = sample_from_bivar_pdf(pdf, N, y, z)
```

Singular Value Decomposition Several functions of this submodule have been used above already. Given a matrix E, we can do the following

```
# computing the largest m singular values and respective vectors
u, s, v = usv_from_E(E, m=3)

# reconstructing an approximate E
E_rank3 = E_from_usv(u, s, v)
```

Depending on the size of E the tuple (u, s, v) can still be large and it may be favourable to fit polynomials through u and v. In ThetaNet we will refer to the set of polynomial coefficients and singular values as "essentials" due the small size of this set.

```
# c_in and c_out being the cluster degrees - gained from a_func_transform()
u_coeff, s, v_coeff = essentials_from_usv(u, s, v, c_in, c_out, deg_k=3)

# compute polynomial approximations
u_poly, s, v_poly = usv_from_essentials(u_coeff, s, v_coeff, c_in, c_out)
```

Eventually, the idea of transforming a matrix A into $E = CAB$ and applying SVD as well as a polynomial fit is packed into the following function:

```
u_coeff_list, s_list, v_coeff_list = essential_list_from_data(A_list, N_c_in,
     N_c_out, deg_k=3, m=3, mapping='cumsum')
```

It is feasible to save these minimal lists together with some more variables which are vital for reconstruction and usage, that is the degree clusters `c_in`, `c_out`, the list of corresponding assortativity values `r_list` and the transformation matrix B.

```
# save
np.savez('svd_list', u=e_list[0], s=e_list[1], v=e_list[2], r_list=r_list,
B=B, c_in=c_in, c_out=c_out)
# load
svd_list = np.load('svd_list.npz')
```

Given the essential lists, we can fit an according set by interpolating coefficients for a certain assortativity.

```
# interpolating essentials for a desired r and computing u,s,v
u_coeff, s, v_coeff = essential_fit(u_coeff_list, s_list, v_coeff_list,
    r_list, r)
```

```
usv = usv_from_essentials(u_coeff, s, v_coeff, c_in, c_out)

# or equivalent
usv_func = tn.utils.usv_func_from_svd_list(svd_list)
usv = usv_func(r)
```

5.5 Example

Suppose we were to investigate the effects of in-/out-degree correlation within a neuron on the network dynamics. That is we want to reproduce Figure 2.6 and Figure 2.7 by utilising the `transform` approach. The files can be found in the example folder of the repository (`https://github.com/cblasche/ThetaNet/tree/master/example`).

parameters.py

We begin with creating a file where we define parameters as in Section 2:

```
import numpy as np
import thetanet as tn

""" Degree space and probability
"""
N = 2000  # number of neurons

k_in_min = 100  # lowest occurring node degree
k_in_max = 400  # highest occurring node degree
k_in = np.arange(k_in_min, k_in_max + 1)
N_k_in = len(k_in)

k_out = np.copy(k_in)
N_k_out = len(k_out)

P_k_in = k_in.astype(float) ** (-3)
P_k_in = P_k_in / np.sum(P_k_in)  # to have sum(P_k)=1
P_k_out = np.copy(P_k_in)

k_mean = np.sum(k_in * P_k_in)  # average value of node degrees


""" Degree network
"""
degree_approach = 'transform'
N_c_in = 10  # number of degree clusters
N_c_out = N_c_in
```

```
""" Neuron dynamics
"""
# coupling strength
kappa = 1.5

# pulse function
n = 2  # sharpness parameter
d_n = 2**n * (np.math.factorial(n)) ** 2 /\
      float(np.math.factorial(2*n))  # normalisation factor
Gamma = tn.dynamics.degree_network.Gamma(n)  # ensemble coefficients

# eta's drawn from Lorentzian ( Cauchy ) distribution
eta_0 = 0  # centre of distribution
delta = 0.05  # width of distribution

# time
t = np.linspace(0, 30, 1000)


""" Continuation
"""
c_var = 'eta_0'  # parameter for single parameter continuation
c_var2 = 'rho'  # additional parameter for bifurcation tracking
c_ds = -0.05  # step size
c_steps = 80  # number of steps
c_n = 7  # number of Newton iterations
c_pmap = False  # continue poincare-map?
```

create_svd_list.py

Since we eventually continue solutions with respect to the correlation ρ, a parameterized connectivity $E(\rho)$, or to be precise its singular value decomposition $u(\rho)$, $s(\rho)$, and $v(\rho)$, will be required. The following file creates a list of decompositions at certain values of ρ and saves it under the name `svd_list.npz`:

```
import parameters as pm
import thetanet as tn
import numpy as np


def main():
    def A_of_rho(rho):
        P_k = tn.utils.bivar_pdf(pm.P_k_in, pm.P_k_out, rho_gauss=rho)
        K_in, K_out = tn.generate.degree_sequence_copula(P_k, pm.N,
                                pm.k_in, pm.k_out, console_output=False)
        A = tn.generate.chung_lu_model(K_in, K_out)
        return A
```

```
    rho_list = np.linspace(-0.99, 0.99, 6)
    A_list = np.asarray([A_of_rho(rho) for rho in rho_list])

    e_list = tn.utils.essential_list_from_data(A_list, pm.N_c_in, pm.N_c_out)
    B, c_in, c_out = tn.generate.a_func_transform(A_list[0], pm.N_c_in,
                                                  pm.N_c_out)[1:]

    np.savez('svd_list', u=e_list[0], s=e_list[1], v=e_list[2],
             r_list=rho_list, B=B, c_in=c_in, c_out=c_out)


if __name__ == '__main__':
    main()
```

eta_0_continuation.py

The next file continues the stable solution for $\eta_0 = 0$ to negative η_0 for three different correlations and then produces a figure comparable to Figure 2.6:

```
import parameters as pm
import thetanet as tn
import numpy as np
from matplotlib import pyplot as plt


def main():
    # make usv_func in parameter.py available
    svd_list = np.load('svd_list.npz')
    pm.usv_func = tn.utils.usv_func_from_svd_list(svd_list)
    pm.B = svd_list['B']

    def continue_and_plot(rho, c):
        # update values in the parameter file
        pm.rho = rho
        pm.usv = pm.usv_func(pm.rho)
        pm.ds = -0.05
        pm.eta_0 = 0

        # run continuation and compute average firing frequency
        b, x, stable = tn.continuation.single_param(pm)
        z = pm.B.dot(b.T).mean(0)
        f = 1 / np.pi * ((1 - z) / (1 + z)).real

        # slicing unstable and stable parts for plots
        sn = (np.where(stable[1:] != stable[:-1])[0] + 1).tolist()
        f = [f[i:j] for i, j in zip([0] + sn, sn + [None])]
        x = [x[i:j] for i, j in zip([0] + sn, sn + [None])]
```

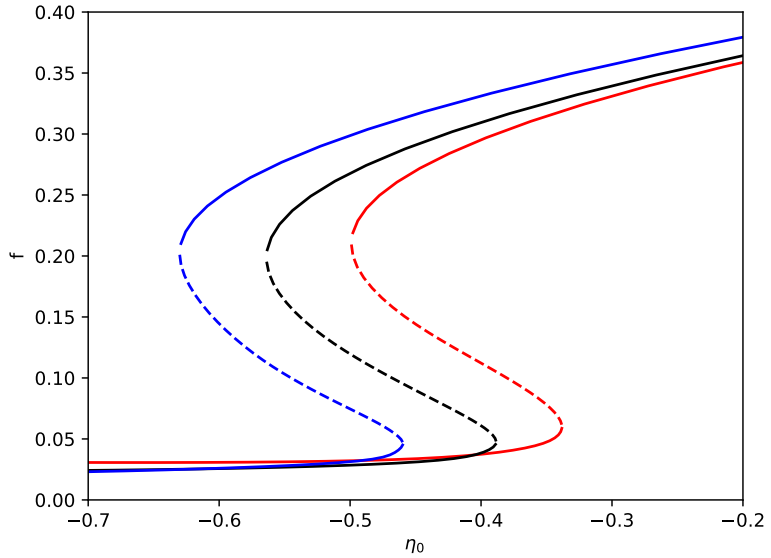

Figure 5.1: This figure shows three continuation curves for (from left to right) $\hat{\rho} = 0.55$, $\hat{\rho} = 0$, and $\hat{\rho} = -0.7$. Using the `transform` approach, results are very well in agreement with Figure 2.6. Note that here we are using $\hat{\rho}$ and not ρ.

```
        stable = [stable[i:j] for i, j in zip([0] + sn, sn + [None])]

        # plot
        ls_dict = {0.: '--', 1.: '-'}
        for ff, xx, ss in zip(f, x, stable):
            plt.plot(xx, ff, ls=ls_dict[ss[0]], c=c)

    # continuation for 3 rho values and plot them in assigned colours
    [continue_and_plot(rho, c) for rho, c in zip([-0.7, 0, 0.55], ['r', 'k',
    'b'])]
    plt.axis([-0.7, -0.2, 0, 0.4])
    plt.xlabel(r'$\eta_0$')
    plt.ylabel('f')
    plt.show()


if __name__ == '__main__':
    main()
```

The result is illustrated in Figure 5.1.

bifurcation_tracking.py

To reproduce Figure 2.7 we run the following code:

```
import parameters as pm
import thetanet as tn
import numpy as np
from matplotlib import pyplot as plt


def main():
    # make usv_func in parameter.py available
    svd_list = np.load('svd_list.npz')
    pm.usv_func = tn.utils.usv_func_from_svd_list(svd_list)

    init_rho = -0.8
    pm.rho = init_rho
    pm.usv = pm.usv_func(pm.rho)

    b, x, stable = tn.continuation.single_param(pm)
    sn = (np.where(stable[1:] != stable[:-1])[0] + 1).tolist()

    def continue_and_plot(i):
        pm.c_ds = 0.05  # continue from init_rho to positive direction
        pm.c_steps = 130
        x_sn, y_sn = tn.continuation.saddle_node(pm, init_b=b[i], init_x=x[i
    ], init_y=init_rho)[1:]
        plt.plot(x_sn, y_sn, c='k')

    [continue_and_plot(i) for i in sn]
    plt.axis([-0.7, -0.3, -0.8, 0.8])
    plt.xlabel(r'$\eta_0$')
    plt.ylabel(r'$\hat{\rho}$')
    plt.show()


if __name__ == '__main__':
    main()
```

The plot is shown in Figure 5.2.

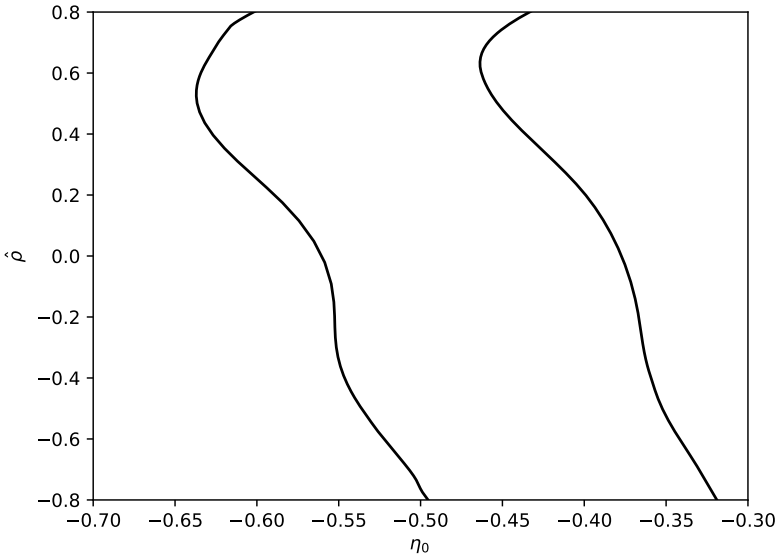

Figure 5.2: We find the results to be similar to Figure 2.7. Again, we are using $\hat{\rho}$ instead of ρ.

Chapter 6

Conclusion and discussion

Networks of neurons form dynamical systems and exhibit activity even in the absence of external stimuli. Novel imaging techniques have made it possible to create neural connectivity maps of small brain regions, yet the interplay between structure and observed neural dynamics remains poorly understood. To advance this understanding, in this thesis we have explored how self-sustained activity changes with degree correlations.

Our investigations employed the theta neuron model and derived degree mean equations using Ott/Antonsen theory. For degree correlations within neurons (Chapter 2), we followed previous studies using a plausible probability function in order to model the the likelihood of connections between respective degrees. This has been done with the intention to reduce the number of variables in our system. That is starting from N neurons one would end up with $K^{\text{in}} \times K^{\text{out}}$ degrees, with K^{in} (K^{out}) being the number of distinct in- (out-) degrees. Depending on the network configuration this can actually increase complexity. However, we identified those restrictions for which the dynamical equations become independent of out-degrees and how the in-degree space can be sampled best. Ultimately, we found that the overall dynamics could be described surprisingly well with a very small number of "virtual degrees". The results we obtained for degree correlation within neurons are in agreement with simulations using a more realistic neuron model, and with other studies: Positive correlation increases the mean firing rate, whereas negative correlation does the opposite.

When turning to degree assortativity in Chapter 3, we found the previous mean field approach to be rather inaccurate, especially for strongly assortative networks. Instead of being based on an approximate assortativity function, our approach is derived from an adjacency matrix. We therefore developed an algorithm to manipulate these matrices in order to increase or decrease assortativity. Translating the system into a degree mean field, the number of variables, again, changes from N to $K^{\text{in}} \times K^{\text{out}}$, but can also be reduced further to a small number of degrees representing a degree cluster, similar to the "virtual degrees" of

Chapter 2. We showed how to parameterize the degree connectivity matrix and applied numerical continuation with respect to assortativity. The mean field derivation we present has proven to be a powerful framework with great potential for application to any feature that can be incorporated in an adjacency matrix. Our investigation in the four different kinds of degree assortativity revealed that (out,in)- and (out,out)-assortativity have no influence at all on the overall dynamics of a neuronal network. For the chosen parameters in this work we observed only minor changes for (in,out)-assortativity whereas (in,in)-assortativity had a stronger impact on the system's dynamics. We have been able to verify mean field results with full theta neuron network simulations, but can hardly relate them to other studies, since degree assortativity in directed networks has not been studied in such isolated conditions before.

Our developed approach of averaging ensemble equations to derive a mean field model can be easily applied to other models or different features, as we shall see below.

Winfree oscillators In this work we only considered type I neurons close to the transition between resting and firing. In a regime far from this point of bifurcation when neurons are constantly firing, they can be modelled as oscillators, e.g. using the Winfree model, which reads for N oscillators

$$\frac{d\theta_j}{dt} = \eta_j + S(\theta_j)\frac{K}{\langle k\rangle}\sum_{n=1}^{N} A_{jn}P(\theta_n) \tag{6.0.1}$$

with $j = 1, \dots, N$ and the phase response curve (PRC)

$$S(\theta) = d_\beta[\sin(\beta) - \sin(\theta+\beta)]; \qquad \beta \in [0, \pi/2] \qquad \text{and} \qquad \int_0^{2\pi} (S(\theta))^2\, d\theta = 1 \tag{6.0.2}$$

For parameter $\beta \in [0, \pi/2)$ the PRC models a type II behaviour, whereas $\beta = \pi/2$ corresponds to type I neurons. The instantaneous coupled pulse of the form

$$P(\theta) = a_q(1-\cos\theta)^q; \qquad q \in \{2, 3, \dots\} \qquad \text{and} \qquad \int_0^{2\pi} P_q(\theta)d\theta = 2\pi \tag{6.0.3}$$

implies an action potential at $\pi = 0$.

When drawing the natural frequencies η_j from a Lorentzian distribution $g(\eta|\eta_0, \Delta)$, we can apply the Ott/Anstonsen ansatz in a similar fashion as in Chapter 3. Consider an infinite ensemble of networks with the same connectivity A_{jn} where each network's set $\{\eta_j\}$

is drawn from $g(\eta)$. We follow [Lai17] and write

$$\frac{dz_j}{dt} = \frac{d_\beta e^{-i\beta}}{2} J_j + \left[i\eta_0 + \Delta + i\sin(\beta) d_\beta J_j\right] z_j - \frac{d_\beta e^{i\beta}}{2} J_j z_j^2 \tag{6.0.4}$$

$$\tag{6.0.5}$$

with

$$J_j = \frac{K}{\langle k \rangle} \sum_{n=1}^{N} A_{jn} a_q \left[\widehat{C}_0 + \sum_{p=1}^{q} \widehat{C}_p \left(z_n^p + \bar{z}_n^p\right)\right] \tag{6.0.6}$$

where $\bar{z}$ is the complex conjugate of z, and

$$\widehat{C_p} = \sum_{k=0}^{q} \sum_{m=0}^{k} \frac{\delta_{k-2m,p} q!}{2^k (q-k)! m! (k-m)!} \tag{6.0.7}$$

The system (6.0.4)-(6.0.6) is amenable to the application of our theory in Section 3.2.2, that is we average all oscillators with the same degree. Let $\boldsymbol{z}$ be the vector of all oscillator ensemble states z_j, b_s the average of all z_j sharing the same degree, and $\boldsymbol{b}$ the respective vector. The index s runs through all $N_{\boldsymbol{k}}$ distinct in- and out-degree tuples $\boldsymbol{k}$ of the network. We can formally write $\boldsymbol{b} = C\boldsymbol{z}$ and thus the dynamical equations are

$$\frac{db_s}{dt} = \frac{d_\beta e^{-i\beta}}{2} \tilde{J}_s + \left[i\eta_0 + \Delta + i\sin(\beta) d_\beta \tilde{J}_s\right] b_s - \frac{d_\beta e^{i\beta}}{2} \tilde{J}_s b_s^2 \tag{6.0.8}$$

$$\tag{6.0.9}$$

with

$$\tilde{J}_s = \frac{K}{\langle k \rangle} \sum_{t=1}^{N_{\boldsymbol{k}}} E_{st} a_q \left[\widehat{C}_0 + \sum_{p=1}^{q} \widehat{C}_p \left(b_t^p + \bar{b}_t^p\right)\right] \tag{6.0.10}$$

where $E = CAB$ and $CB = I_{N_{\boldsymbol{k}}}$, the $N_{\boldsymbol{k}} \times N_{\boldsymbol{k}}$ identity matrix. Thus, we can simulate a network of Winfree oscillators with a given adjacency matrix using a reduced set of mean field equations.

Gap junction coupling In addition to synaptic coupling, neurons are known to interact through gap-junctions as well. That is a neuron's membrane voltage is influenced by a near-by neuron. This type of connectivity can be incorporated in a network of N theta neurons as follows [Lai15]:

$$\frac{d\theta_j}{dt} = 1 - \cos\theta_j - \kappa\sin(\theta_j) + (1 + \cos\theta_j)(\eta_j + \frac{\kappa}{\langle k \rangle} \sum_{n=1}^{N} \Lambda_{jn} q(\theta_n)) \tag{6.0.11}$$

with

$$q(\theta) = \frac{\sin(\theta)}{1+\cos(\theta)+\epsilon} \qquad \text{and} \qquad 0 < \epsilon \ll 1 \tag{6.0.12}$$

where $j = 1, \ldots, N$ and κ is the gap junction coupling strength. The function $q(\theta)$ approximates the membrane voltage $\tan(\theta/2)$ of a QIF neuron. The adjacency matrix Λ_{jn} describes which neurons are close enough together to be coupled through gap junctions. Since gap junctions depend only on absolute distance, we find this type of network to be undirected and neurons are described only by a degree k instead of a tuple of in- and out-degree. Again, we can make use of the Ott/Antonsen ansatz and derive ensemble equations if the set $\{\eta_j\}$ is drawn from a Lorentzian $g(\eta|\eta_0, \Delta)$ [Lai15]:

$$\frac{dz_j}{dt} = \frac{(\kappa - i)(z_j - 1)^2}{2} + \frac{i(z_j+1)^2}{2}\left[\eta_0 - i\Delta + \frac{\kappa}{\langle k\rangle}\sum_{n=1}^{N}\Lambda_{jn}Q(z_n)\right] \tag{6.0.13}$$

$$\tag{6.0.14}$$

with

$$Q(z_n) = \sum_{m=1}^{\infty}\left(c_m z_n^m + \bar{c}_m \bar{z}_n^m\right) \tag{6.0.15}$$

where

$$c_m = \frac{i\left(\rho^{m+1} - \rho^{m-1}\right)}{2(\rho + 1 + \epsilon)} \qquad \text{and} \qquad \rho = \sqrt{2\epsilon + \epsilon^2} - 1 - \epsilon \tag{6.0.16}$$

Again, we compute the variable b_s by averaging all ensemble variables z_j sharing the same degree k, where s is the index of k in the discrete degree space. With $\boldsymbol{b}$ and $\boldsymbol{z}$ being the respective vectors we have $\boldsymbol{b} = C\boldsymbol{z}$ and arrive at

$$\frac{db_s}{dt} = \frac{(\kappa - i)(b_s - 1)^2}{2} + \frac{i(b_s+1)^2}{2}\left[\eta_0 - i\Delta + \frac{\kappa}{\langle k\rangle}\sum_{t=1}^{N_k}\mathcal{E}_{st}Q(b_t)\right] \tag{6.0.17}$$

$$\tag{6.0.18}$$

with

$$Q(b_t) = \sum_{m=1}^{\infty}\left(c_m b_t^m + \bar{c}_m \bar{b}_t^m\right) \tag{6.0.19}$$

and s running through the N_k indices of distinct degrees k. The degree connectivity is $\mathcal{E} = C\Lambda B$ and $CB = I_{N_{\boldsymbol{k}}}$.

Those two examples illustrate the straightforward application of the methods we developed in Section 3.2. Despite evidence of neuronal networks comprising both inhibitory and excitatory connections, we decided to simplify the model network to assume that it is composed of purely the one type or the other. Incorporating a network with both populations interacting can be done in a straightforward manner using these frameworks. A future study could formulate an excitability mean field, or a mixture of both. Thus, structural network features with respect to those properties could be investigated in a very efficient manner.

References

[AB02] R. Albert and A. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47–97, 2002.

[Abb99] L. Abbott. Lapicque's introduction of the integrate-and-fire model neuron (1907). *Brain Research Bulletin*, 50(5):303 – 304, 1999.

[AGAP+12] V. Avalos-Gaytán, J. Almendral, D. Papo, S. Schaeffer, and S. Boccaletti. Assortative and modular networks are shaped by adaptive synchronization processes. *Physical Review E*, 86(1), 2012.

[BAO11] G. Barlev, T. M. Antonsen, and E. Ott. The dynamics of network coupled phase oscillators: An ensemble approach. *Chaos*, 21(2):025103, 2011.

[BML20] C. Bläsche, S. Mean, and C. Laing. Degree assortativity in networks of spiking neurons. *Journal of Computational Dynamics*, 7(2):401–423, 2020.

[CB19] S. Coombes and Á. Byrne. Next generation neural mass models. In *Nonlinear Dynamics in Computational Neuroscience*, pages 1–16. Springer, 2019.

[CGDM13] B. Coutinho, A. Goltsev, S. Dorogovtsev, and J. Mendes. Kuramoto model with frequency-degree correlations on complex networks. *Physical Review E*, 87(3):032106, 2013.

[CHC+17] S. Chandra, D. Hathcock, K. Crain, T. M. Antonsen, M. Girvan, and E. Ott. Modeling the network dynamics of pulse-coupled neurons. *Chaos*, 27(3):033102, 2017.

[CL90] S.-N. Chow and X.-B. Lin. Bifurcation of a homoclinic orbit with a saddle-node equilibrium. *Differential and Integral Equations*, 3:435–466, 1990.

[CL02] F. Chung and L. Lu. Connected components in random graphs with given expected degree sequences. *Annals of combinatorics*, 6(2):125–145, 2002.

[CSK+18] G. Collin, L. J. Seidman, M. S. Keshavan, W. S. Stone, Z. Qi, T. Zhang, Y. Tang, H. Li, S. A. Anteraper, M. A. Niznikiewicz, et al. Functional connectome organization predicts conversion to psychosis in clinical high-risk youth from the sharp program. *Molecular psychiatry*, pages 1–10, 2018.

[DFJT11] S. De Franciscis, S. Johnson, and J. Torres. Enhancing neural-network performance via assortativity. *Physical Review E*, 83(3), 2011.

[dSSSNL+14] D. de Santos-Sierra, I. Sendiña-Nadal, I. Leyva, J. A. Almendral, S. Anava, A. Ayali, D. Papo, and S. Boccaletti. Emergence of small-world anatomical networks in self-organizing clustered neuronal cultures. *PLoS One*, 9(1):e85828, 2014.

[ECC+05] V. M. Eguíluz, D. R. Chialvo, G. A. Cecchi, M. Baliki, and A. V. Apkarian. Scale-free brain functional networks. *Physical Review Letters*, 94:018102, Jan 2005.

[EK86] B. Ermentrout and N. Kopell. Parabolic bursting in an excitable system coupled with a slow oscillation. *SIAM Journal on Applied Mathematics*, 46(2):233–253, 1986.

[EK90] B. Ermentrout and N. Kopell. Oscillator death in systems of coupled neural oscillators. *SIAM Journal on Applied Mathematics*, 50(1):125–146, 1990.

[Eng06] S. Engblom. Gaussian quadratures with respect to discrete measures. Technical report, Uppsala University, Technical Report 2006-007, 2006.

[Erm96] B. Ermentrout. Type I membranes, phase resetting curves, and synchrony. *Neural Computation*, 8(5):979–1001, 1996.

[EUA+12] D. V. Essen, K. Ugurbil, E. Auerbach, D. Barch, T. Behrens, R. Bucholz, et al. The Human Connectome Project: a data acquisition perspective. *NeuroImage*, 62(4):2222–2231, 2012.

[FFGP10] J. G. Foster, D. V. Foster, P. Grassberger, and M. Paczuski. Edge direction and the structure of networks. *Proceedings of the National Academy of Sciences*, 107(24):10815–10820, 2010.

[FT13] N. Fourcaud-Trocmé. *Integrate and Fire Models, Deterministic*, pages 1–9. Springer New York, New York, NY, 2013.

[FTHVVB03] N. Fourcaud-Trocmé, D. Hansel, C. Van Vreeswijk, and N. Brunel. How spike generation mechanisms determine the neuronal response to fluctuating inputs. *Journal of Neuroscience*, 23(37):11628–11640, 2003.

[Gov00] W. J. Govaerts. *Numerical methods for bifurcations of dynamical equilibria*, volume 66. Siam, Philadelphia, 2000.

[GR71] G. H. Golub and C. Reinsch. Singular value decomposition and least squares solutions. In *Linear Algebra*, volume 14, pages 134–151. Springer, apr 1971.

[GW69] G. H. Golub and J. H. Welsch. Calculation of gauss quadrature rules. *Mathematics of computation*, 23(106):221–230, 1969.

[HCG+08] P. Hagmann, L. Cammoun, X. Gigandet, R. Meuli, C. Honey, J. Van Wedeen, and O. Sporns. Mapping the structural core of human cerebral cortex. *PLoS Biology*, 6(7):1479–1493, 2008.

[HH52] A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, 117:500–544, 1952.

[HI97] F. C. Hoppensteadt and E. M. Izhikevich. *Weakly Connected Neural Networks*. Springer-Verlag, Berlin, Heidelberg, 1997.

[HTJSB13] Y. Hu, J. Trousdale, K. Josić, and E. Shea-Brown. Motif statistics and spike correlations in neuronal networks. *Journal of Statistical Mechanics: Theory and Experiment*, (03):P03012, 2013.

[Ich04] T. Ichinomiya. Frequency synchronization in a random oscillator network. *Physical Review E*, 70(2):026116, 2004.

[Izh03] E. M. Izhikevich. Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6):1569–1572, 2003.

[Izh07] E. M. Izhikevich. *Dynamical Systems in Neuroscience : The Geometry of Excitability and Bursting.* Computational Neuroscience. The MIT Press, 2007.

[JK17] E. Jonas and K. P. Kording. Could a neuroscientist understand a microprocessor? *PLoS Computational Biology*, 13(1):1–24, 2017.

[KSR17] M. Kähne, I. Sokolov, and S. Rüdiger. Population equations for degree-heterogenous neural networks. *Physical Review E*, 96(5):052306, 2017.

[Lai14a] C. R. Laing. Derivation of a neural field model from a network of theta neurons. *Physical Review E*, 90(1):010901, 2014.

[Lai14b] C. R. Laing. Numerical bifurcation theory for high-dimensional neural models. *The Journal of Mathematical Neuroscience*, 4(1):13, 2014.

[Lai15] C. R. Laing. Exact neural fields incorporating gap junctions. *SIAM Journal on Applied Dynamical Systems*, 14(4):1899–1929, 2015.

[Lai16] C. R. Laing. Bumps in small-world networks. *Frontiers in Computational Neuroscience*, 10:53, 2016.

[Lai17] C. R. Laing. *Phase Oscillator Network Models of Brain Dynamics*, chapter 37, pages 505–517. John Wiley & Sons, Ltd, 2017.

[LB20] C. Laing and C. Bläsche. The effects of within-neuron degree correlations in networks of spiking neurons. *Biological Cybernetics*, 114:337–347, 2020.

[LBS13] T. B. Luke, E. Barreto, and P. So. Complete classification of the macroscopic behavior of a heterogeneous network of theta neurons. *Neural computation*, 25(12):3207–3234, 2013.

[LS10] M. D. LaMar and G. D. Smith. Effect of node-degree correlation on synchronization of identical pulse-coupled oscillators. *Physical Review E*, 81(4):046206, 11, 2010.

[Mar06] H. Markram. The blue brain project. In *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, SC '06, page 53–es, New York, NY, USA, 2006. Association for Computing Machinery.

[MBL20] S. Means, C. Bläsche, and C. Laing. A permutation method for network assembly. *To appear in PLoS One*, 2020.

[MHT17] M. B. Martens, A. R. Houweling, and P. H. Tiesinga. Anti-correlations in the degree distribution increase stimulus detection performance in noisy spiking neural networks. *Journal of Computational Neuroscience*, 42(1):87–106, 2017.

[ML81] C. Morris and H. Lecar. Voltage oscillations in the barnacle giant muscle fiber. *Biophysical Journal*, 35(1):193 – 213, 1981.

[MMFDCa14] L. A. A. Meira, V. R. Máximo, A. L. Fazenda, and A. F. Da Conceição. Acc-motif: Accelerated network motif detection. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 11(5):853–862, 2014.

[MP48] W. S. McCulloch and W. Pitts. The statistical organization of nervous activity. *Biometrics*, 4(2):91—-99, 1948.

[MPR15] E. Montbrió, D. Pazó, and A. Roxin. Macroscopic description for networks of spiking neurons. *Physical Review X*, 5(2):021028, 2015.

[MS09] S. A. Marvel and S. H. Strogatz. Invariant submanifold for series arrays of Josephson junctions. *Chaos*, 19(1):013132, 2009.

[Nel07] R. B. Nelsen. *An introduction to copulas*. Springer Science & Business Media, 2007.

[New02] M. E. J. Newman. Assortative mixing in networks. *Physical Review Letters*, 89:208701, 2002.

[New03] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

[NFS+17] D. Q. Nykamp, D. Friedman, S. Shaker, M. Shinn, M. Vella, A. Compte, and A. Roxin. Mean-field equations for neuronal networks with arbitrary degree distributions. *Physical Review E*, 95:042323, 2017.

[OA08] E. Ott and T. M. Antonsen. Low dimensional behavior of large systems of globally coupled oscillators. *Chaos*, 18(3):037113, 2008.

[OA09] E. Ott and T. M. Antonsen. Long time evolution of phase oscillator systems. *Chaos*, 19:023117, 2009.

[OLKD15] G. K. Ocker, A. Litwin-Kumar, and B. Doiron. Self-organization of microcircuits in networks of spiking neurons with plastic synapses. *PLoS Computational Biology*, 11(8):e1004458, 2015.

[PBM11] R. Perin, T. K. Berger, and H. Markram. A synaptic organizing principle for cortical neuronal groups. *Proceedings of the National Academy of Sciences*, 108(13):5419–5424, 2011.

[Pet19] F. Peter. *Transition to synchrony in finite Kuramoto ensembles*. doctoralthesis, Universität Potsdam, 2019.

[RO14] J. G. Restrepo and E. Ott. Mean-field theory of assortative networks of phase oscillators. *Europhysics Letters*, 107(6):60006, 2014.

[Rox11] A. Roxin. The role of degree distribution in shaping the dynamics in networks of sparsely connected spiking neurons. *Frontiers in Computational Neuroscience*, 5:8, 2011.

[RS10] M. Rubinov and O. Sporns. Complex network measures of brain connectivity: uses and interpretations. *Neuroimage*, 52(3):1059–1069, 2010.

[SBRP+02] A. Salinger, N. Bou-Rabee, R. Pawlowski, E. Wilkes, E. Burroughs, R. Lehoucq, and L. Romero. Loca 1.0 library of continuation algorithms: theory and implementation manual. *Sandia National Laboratories, SAND2002-0396*, 2002.

[Seu12] S. Seung. *Connectome: How the brain's wiring makes us who we are.* Houghton Mifflin Harcourt, 2012.

[SHZ+13] B. Schwab, T. Heida, Y. Zhao, E. Marani, S. van Gils, and R. Van Wezel. Synchrony in parkinson's disease: importance of intrinsic properties of the external globus pallidus. *Frontiers in Systems Neuroscience*, 7:60, 2013.

[SKSR15] C. Schmeltzer, A. H. Kihara, I. M. Sokolov, and S. Rüdiger. Degree correlations optimize neuronal network sensitivity to sub-threshold stimuli. *PLoS One*, 10:e0121794, 2015.

[SLB14] P. So, T. B. Luke, and E. Barreto. Networks of theta neurons with time-varying excitability: Macroscopic chaos, multistability, and final-state uncertainty. *Physica D: Nonlinear Phenomena*, 267:16–26, 2014.

[SNAD17] L. Swanson, E. Newman, A. Araque, and J. Dubinsky. *The Beautiful Brain: The Drawings of Santiago Ramon y Cajal.* ABRAMS, 2017.

[SOSE+16] C. Stephan-Otto, S. Siddi, J. C. Esteban], C. Senior, R. García-Álvarez, M. R. Cambra-Martí, J. Usall, and G. Brébion. Neural activity during object perception in schizophrenia patients is associated with illness duration and affective symptoms. *Schizophrenia Research*, 175(1):27 – 34, 2016.

[Spo10] O. Sporns. *Networks of the Brain.* MIT press, 2010.

[SRO15] P. S. Skardal, J. G. Restrepo, and E. Ott. Frequency assortativity can induce chaos in oscillator networks. *Physical Review E*, 91(6):060902, 2015.

[SSR+05] S. Song, P. J. Sjöström, M. Reigl, S. Nelson, and D. B. Chklovskii. Highly nonrandom features of synaptic connectivity in local cortical circuits. *PLoS Biology*, 3(3):e68, 2005.

[SSSG13] B. Sonnenschein, F. Sagués, and L. Schimansky-Geier. Networks of noisy oscillators with correlated degree and frequency dispersion. *The European Physical Journal B*, 86(1):12, 2013.

[SSTR13] P. S. Skardal, J. Sun, D. Taylor, and J. G. Restrepo. Effects of degree-frequency correlations on network synchronization: Universality and full phase-locking. *Europhysics Letters*, 101(2):20001, 2013.

[TGDD+14] S. Teller, C. Granell, M. De Domenico, J. Soriano, S. Gómez, and A. Arenas. Emergence of assortative mixing between clusters of cultured neurons. *PLoS Computational Biology*, 10(9), 2014.

[TIM+14] N. Timme, S. Ito, M. Myroshnychenko, F.-C. Yeh, E. Hiolski, P. Hottowy, and J. M. Beggs. Multiplex networks of cortical and hippocampal neurons revealed at different timescales. *PLoS One*, 9(12):1–43, 2014.

[TKY+06] K. Tsumoto, H. Kitajima, T. Yoshinaga, K. Aihara, and H. Kawakami. Bifurcations in morris–lecar neuron model. *Neurocomputing*, 69(4-6):293–316, 2006.

[VHT13] J. C. Vasquez, A. R. Houweling, and P. Tiesinga. Simultaneous stability and sensitivity in model cortical networks is achieved through anti-correlations between the in- and out-degree of connectivity. *Frontiers in Computational Neuroscience*, 7:156, 2013.

[VPR17] M. Vegué, R. Perin, and A. Roxin. On the structure of cortical microcircuits inferred from small sample sizes. *Journal of Neuroscience*, 37(35):8498–8510, 2017.

[VR19] M. Vegué and A. Roxin. Firing rate distributions in spiking networks with heterogeneous connectivity. *Physical Review E*, 100:022208, 2019.

[Wat91] D. S. Watkins. *Fundamentals of Matrix Computations*. John Wiley & Sons, Inc., USA, 1991.

[ZBNN11] L. Zhao, B. I. Beverlin, T. Netoff, and D. Q. Nykamp. Synchronization from second order network connectivity statistics. *Frontiers in Computational Neuroscience*, 5:28, 2011.

Appendix A

Matrix creation

A.1 Configuration model

The configuration model is a rather precise and deterministic way of constructing an adjacency matrix. Consider a directed network of N nodes with degrees $\boldsymbol{k} = (k^{\text{in}}, k^{\text{out}})$ and probability $P(\boldsymbol{k})$. Sampling N tuples from $P(\boldsymbol{k})$ leads us to the starting point: the degree sequence $((k_i^{\text{in}}, k_i^{\text{out}}))_{i=1}^{N}$.

In order to create edges, we provide a list of ingoing half-edges or stubs and another list of outgoing counterparts, each of length N_e. Let us denote those lists as sequences $(i_e^{\text{in}})_{e=1}^{N_e}$ and $(i_e^{\text{out}})_{e=1}^{N_e}$, where i_e^{in} is the node index of the incoming side of edge e and outgoing for i_e^{out}, respectively. To construct $(i_e^{\text{in}})_{e=1}^{N_e}$ we simply attach N sequences each consisting of the k_i^{in} repeated value i and equivalent for out-degrees:

$$(i_e^{\text{in}})_{e=1}^{N_e} = \left((i)_{j=1}^{k_i^{\text{in}}}\right)_{i=1}^{N} \qquad \text{and} \qquad (i_e^{\text{out}})_{e=1}^{N_e} = \left((i)_{j=1}^{k_i^{\text{out}}}\right)_{i=1}^{N}. \tag{A.1.1}$$

We form edges by pairing up members of those sequences into one sequence of tuples $((i_e^{\text{in}}, i_e^{\text{out}}))_{e=1}^{N_e}$. Note that one sequence needs to be shuffled, in order to avoid any kind of correlations.

We were already assuming that each stub sequence has length N_e, which is not necessarily the case. Sampling from $P(\boldsymbol{k})$ will only provide two sequences, where $N_e^{\text{in}} = \sum_{i=1}^{N} k_i^{\text{in}}$ and $N_e^{\text{out}} = \sum_{i=1}^{N} k_i^{\text{out}}$ might be close, but not equal. For the configuration model it is necessary to have a matching number of in- and out-stubs. We can insure this by sampling again and again, until we draw two sequences with equal sums. This process can be tedious for large numbers of N. Alternatively, if the difference $d = |N_e^{\text{in}} - N_e^{\text{out}}|$ falls below a threshold, say some fraction of N, we deterministically modify both sequences by picking $d/2$ nodes in both in- and out-degree sequence and lower, or increase respectively, their degree by 1.

Be aware that nodes with minimal or maximal degree should not be chosen to perform this correction to ensure they are still within the degree space. The adjacency matrix A can be constructed straightforwardly from the edge sequence by adding 1 for each edge $(i_e^{\text{in}}, i_e^{\text{out}})$ to the matrix entry $A(i_e^{\text{in}}, i_e^{\text{out}})$.

A.2 Chung Lu model and degree assortativity

The Chung Lu approach is probabilistic, easy to implement and efficient. In addition, the sums of in- and out-degree sequence do not have to be equal and one can introduce assortativity straight away. But all these desirable assets come at a price. The degree distribution in the final network only roughly resembles the initial degree probability $P(\boldsymbol{k})$ and it gets altered further the more degree assortative we want the network to be.

In order to construct an adjacency matrix from the degree sequence $((k_i^{\text{in}}, k_i^{\text{out}}))_{i=1}^N$, we compute a target matrix holding the probabilities to connect nodes based on their degrees, which reads

$$T_{ij} = \frac{k_i^{\text{in}} k_j^{\text{out}}}{N \langle k \rangle}, \tag{A.2.1}$$

with N being the number of nodes and $\langle k \rangle$ being the mean degree. Comparison with a N-by-N uniform random matrix $U_{ij} \in [0, 1]$ yields the adjacency matrix

$$A_{ij} = \begin{cases} 1, & \text{if } T_{ij} > U_{ij} \text{ and} \\ 0, & \text{otherwise.} \end{cases} \tag{A.2.2}$$

In this manner adjacency matrices of neutral assortativity can be constructed very efficiently. The authors of [CHC+17] propose a modified target matrix according to their assortativity function. Sticking to the convention used in the assortativity definition (3.3.2), i.e. $\alpha, \beta \in [\text{in}, \text{out}]$, we implement (α, β) assortativity by simply adding a scalable term and have

$$T_{ij} = \frac{k_i^{\text{in}} k_j^{\text{out}} + c \cdot \left(k_j^{\alpha} - \langle {}^s k^{\alpha} \rangle\right) \left(k_i^{\beta} - \langle {}^r k^{\beta} \rangle\right)}{N \langle k \rangle}. \tag{A.2.3}$$

The mean value over edges poses an irritating issue, since there are no edges at this stage. One solution could be to create a neutral assortative adjacency matrix first and use it to compute those mean values. Another option is to compute them from the degree distributions directly. Consider that the number of available edges is $N_e = N \langle k \rangle$ where $\langle k \rangle$

is given by

$$\langle k \rangle = \sum_{i=1}^{N_{k^{\text{in}}}} P^{\text{in}}(k_i^{\text{in}}) k_i^{\text{in}} = \sum_{i=1}^{N_{k^{\text{out}}}} P^{\text{out}}(k_i^{\text{out}}) k_i^{\text{out}} \tag{A.2.4}$$

and marginal distributions are projections of $P(\boldsymbol{k})$

$$P^{\text{in}}(k^{\text{in}}) = \sum_{i=1}^{N_{k^{\text{out}}}} P(k^{\text{in}}, k_i^{\text{out}}); \qquad P^{\text{out}}(k^{\text{out}}) = \sum_{i=1}^{N_{k^{\text{in}}}} P(k_i^{\text{in}}, k^{\text{out}}). \tag{A.2.5}$$

We compute the mean values by going through all edges, while counting, say all in-degrees of the receiving nodes and subsequently dividing by the number of edges. This can be written as

$$\left\langle {}^{r}k^{\text{in}} \right\rangle = \frac{\sum_{e=1}^{N_e} {}^{r}k_e^{\text{in}}}{N_e}$$

Having $NP^{\text{in}}(k_i^{\text{in}})$ nodes in the network with in-degree k_i^{in}, means we will find k_i^{in} edges for each of them, being in total $NP^{\text{in}}(k_i^{\text{in}})k_i^{\text{in}}$. For every edge of that kind we add k_i^{in} to the count, thus we write

$$\begin{aligned} &= \frac{\sum_{i=1}^{N_{k^{\text{in}}}} NP^{\text{in}}(k_i^{\text{in}}) k_i^{\text{in}} \cdot k_i^{\text{in}}}{N \langle k \rangle} \\ &= \frac{\sum_{i=1}^{N_{k^{\text{in}}}} P^{\text{in}}(k_i^{\text{in}}) \cdot (k_i^{\text{in}})^2}{\langle k \rangle}. \end{aligned} \tag{A.2.6}$$

For the out-degrees of the receiving nodes we have

$$\begin{aligned} \left\langle {}^{r}k^{\text{out}} \right\rangle &= \frac{\sum_{e=1}^{N_e} {}^{r}k_e^{\text{out}}}{N_e} \\ &= \frac{\sum_{i=1}^{N_{k^{\text{in}}}} P^{\text{in}}(k_i^{\text{in}}) k_i^{\text{in}} \cdot k_i^{\text{out}}}{\langle k \rangle}, \end{aligned} \tag{A.2.7}$$

where k_i^{out} has to be expressed in terms of k_i^{in}, thus

$$k_i^{\text{out}} = \frac{\sum_{j=1}^{N_{k^{\text{out}}}} P(k_i^{\text{in}}, k_j^{\text{out}}) k_j^{\text{out}}}{P^{\text{in}}(k_i^{\text{in}})}. \tag{A.2.8}$$

Inserting (A.2.8) in (A.2.7) results in

$$\left\langle {}^{r}k^{\text{out}} \right\rangle = \frac{\sum_{i=1}^{N_{k^{\text{in}}}} \sum_{j=1}^{N_{k^{\text{out}}}} P(k_i^{\text{in}}, k_j^{\text{out}}) k_i^{\text{in}} k_j^{\text{out}}}{\langle k \rangle}. \tag{A.2.9}$$

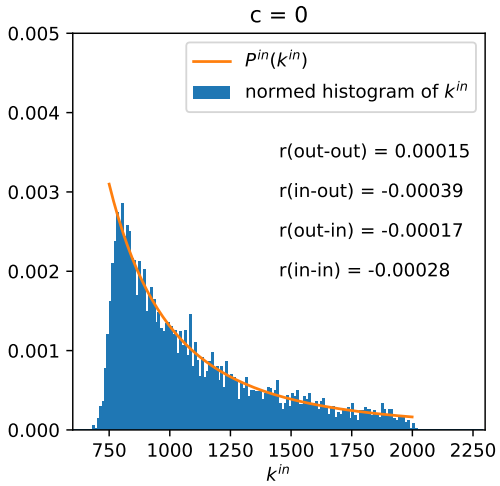

Figure A.1: Comparison of the target in-degree probability and the normalised histogram of in-degrees in a network constructed using the Chung Lu method. Parameters are: $k^{\text{in}} \in [750, 2000]$ and $P^{\text{in}}(k^{\text{in}}) \sim k^{-3}$.

Equivalently, the edge mean values of sending nodes read

$$\left\langle {}^{s}k^{\text{out}} \right\rangle = \frac{\sum_{i=1}^{N_{k^{\text{out}}}} P^{\text{out}}(k_i^{\text{out}}) \cdot (k_i^{\text{out}})^2}{\langle k \rangle} \tag{A.2.10}$$

$$\left\langle {}^{s}k^{\text{in}} \right\rangle = \frac{\sum_{i=1}^{N_{k^{\text{out}}}} \sum_{j=1}^{N_{k^{\text{in}}}} P(k_j^{\text{in}}, k_i^{\text{out}}) k_j^{\text{in}} k_i^{\text{out}}}{\langle k \rangle}. \tag{A.2.11}$$

Note that (A.2.9) and (A.2.11) are equal and thus $\langle {}^{r}k^{\text{out}} \rangle = \left\langle {}^{s}k^{\text{in}} \right\rangle$. Furthermore, without node correlation we have $P(k^{\text{in}}, k^{\text{out}}) = P^{\text{in}}(k^{\text{in}}) P^{\text{out}}(k^{\text{out}})$ and all four mean values are $\langle k \rangle$.

As mentioned earlier, the downside of this approach is that the degree distributions only roughly capture the initial degree probability, see Figure A.1 for the case $c = 0$ – meaning no degree assortativity. Introducing assortativity, we find that the distribution gets altered significantly. An example for (out,in) assortativity is given in Figure A.2.

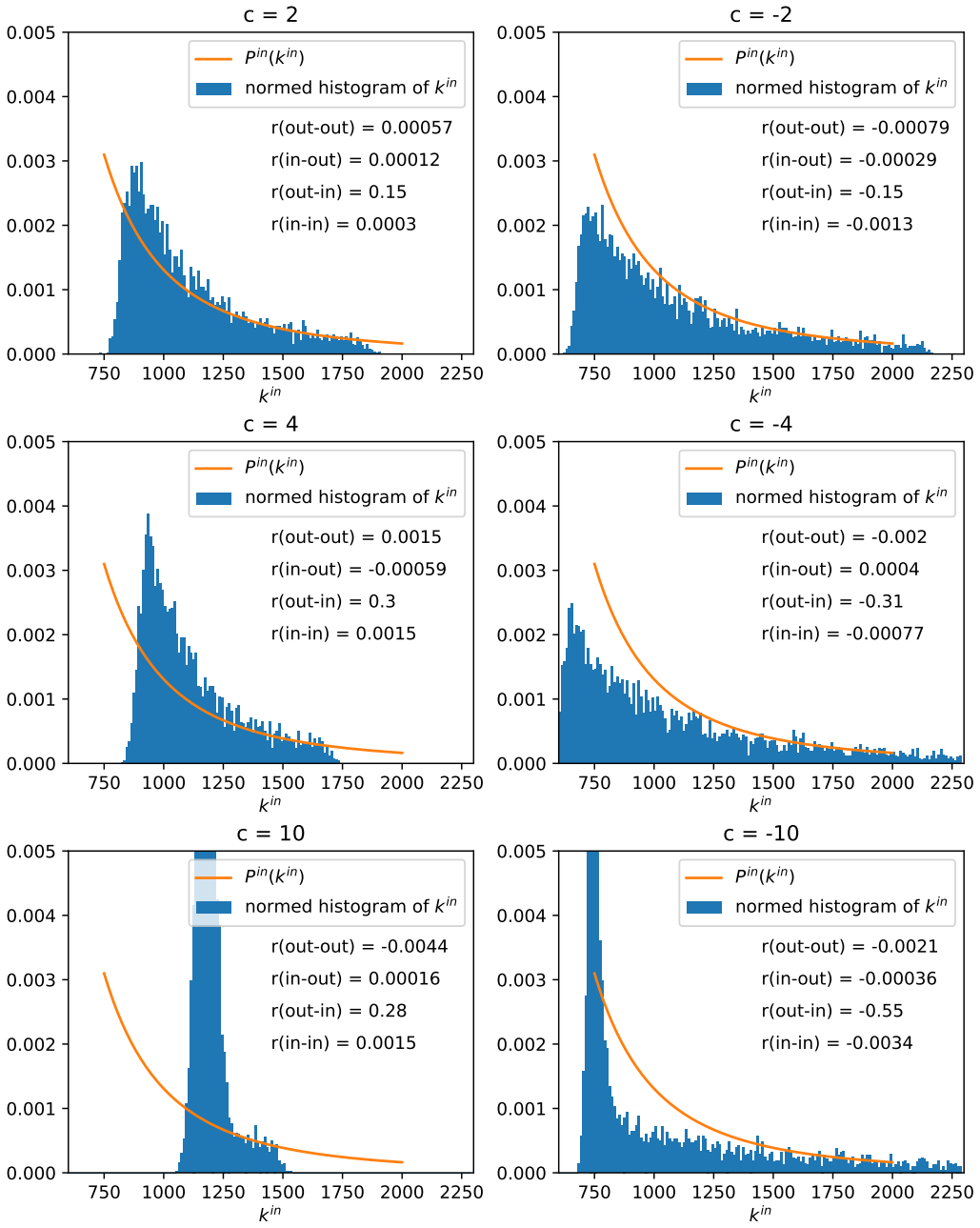

Figure A.2: When introducing degree assortativity using the Chung Lu method, we find the assortativity coefficients $r(\alpha, \beta)$ are altered only in the desired component, (out,in) in this example. Albeit, the degree distributions become significantly disturbed the larger we set the tuning parameter c. Parameters are chosen as in Figure A.1.

Appendix B

Applying Ott/Antonsen theory to theta neurons

Luke et al.[LBS13] have demonstrated how the Ott/Antonsen ansatz can be applied to an all-to-all coupled network of N heterogeneous theta neurons in the limit of $N \to +\infty$. In this work we have made use of their theory twice and derived two different frameworks for structured neuronal network: one following Chandra et al. [CHC+17] and one using an ensemble approach (Section 3.2). Below, we want to follow the calculations for the latter in more detail.

B.1 The ansatz and the system

Given a heterogeneous network structure, we find it useful to consider an infinite ensemble of networks, where each member has the same adjacency matrix [BAO11]. The neuronal networks within this ensemble differ from another with respect to their intrinsic excitability η, i.e. the excitability η_j of neuron j across the ensemble is captured in the distribution $g(\eta_j)$. In this infinite limit we can describe neuron j with a probability function $f_j(\theta_j, \eta_j, t)$, which captures the likelihood to have the intrinsic excitability η_j and to be in the state θ_j at time t. The time evolution of this probability is governed by the continuity equation

$$\frac{\partial f_j(\theta_j, \eta_j, t)}{\partial t} + \frac{\partial}{\partial \theta_j} \{v_j f_j(\theta_j, \eta_j, t)\} = 0 \tag{B.1.1}$$

where the velocity field v_j of the neuronal state θ_j is the continuous version of a theta neuron and reads

$$v_j = 1 - \cos\theta_j + (1 + \cos\theta_j)(\eta_j + I_j) \tag{B.1.2}$$

with synaptic current

$$I_j(t) = \frac{K}{\langle k \rangle} \sum_{n=1}^{N} A_{jn} \int_{-\infty}^{+\infty} \int_{0}^{2\pi} P_q(\theta_n) f_n(\theta_n, \eta_n, t) d\theta_n d\eta_n \tag{B.1.3}$$

Ott and Antonsen [OA08] studied a system of all-to-all coupled Kuramoto oscillators and were able to solve the continuity equation using a Poisson kernel as probability function. This ansatz works for theta neurons as well and is typically written in the form of a Fourier series

$$f_j(\theta_j, \eta_j, t) = \frac{g(\eta_j)}{2\pi} \left[1 + \sum_{k=1}^{\infty} \{\alpha_j(\eta_j, t)\}^k e^{ik\theta_j} + \sum_{k=1}^{\infty} \{\bar{\alpha}_j(\eta_j, t)\}^k e^{-ik\theta_j} \right] \tag{B.1.4}$$

where $\bar{\alpha}$ denotes the complex conjugate of α. To ensure convergence of the series, the coefficients are required to fulfill $|\alpha_j(\eta_j, t)| \leqslant 1$. For each neuron, the θ dependence is given by two parameters, the real and the imaginary part of $\alpha(\eta_j, t)$. In assuming Eq. (B.1.4), we restrict a general Fourier series with arbitrary coefficients to a very special form: the kth coefficient is the kth power of some complex function $\alpha(\eta_j, t)$. Below, we show under which condition this two-dimensional sub-manifold is invariant.

B.2 Synaptic current in an ensemble

In Eq. (B.1.3) we integrate over the pulse function $P_q(\theta_n)$ which we have introduced in Eq.1.2.13. The parameter q models the pulse's sharpness. In the mean field, we weight this function with probability $f_n(\theta_n, \eta_n, t)$ and compute the mean field pulse function $H(\alpha_n(\eta_n, t); q)$

$$H(\alpha_n(\eta_n, t); q) = \int_{-\infty}^{+\infty} \int_{0}^{2\pi} a_q (1 - \cos\theta_n)^q \frac{g(\eta_n)}{2\pi} \left[1 + \sum_{k=1}^{\infty} \{\alpha_n(\eta_n, t)\}^k e^{ik\theta_n} + c.c. \right] d\theta_n d\eta_n \tag{B.2.1}$$

where *c.c.* denotes the complex conjugate of the previous term. We follow So et al.[SLB14], apply the binomial theorem twice, and rewrite

$$(1 - \cos\theta_n)^q = \sum_{l=0}^{q} \sum_{m=0}^{l} D_{lm} e^{i(l-2m)\theta_n} \tag{B.2.2}$$

with

$$D_{lm} = \frac{q!(-1)^l}{2^l (q-l)! m! (l-m)!} \tag{B.2.3}$$

This allows the integration over θ_n in Eq. (B.2.1):

$$H(\alpha_n(\eta_n,t);q) = \int_{-\infty}^{+\infty} g(\eta_n)a_q \left[C_0 + \sum_{r=1}^{q} C_r\{\alpha_n(\eta_n,t)\}^r + c.c. \right] d\eta_n \tag{B.2.4}$$

where

$$C_r = \sum_{l=0}^{q} \sum_{m=0}^{l} \delta_{l-2m,r} D_{lm} \tag{B.2.5}$$

By choosing the intrinsic excitability of neuron n across the ensemble to be drawn from a Lorentzian distribution $g(\eta_n|\eta_0,\Delta)$ with mean η_0 and half-width at half-maximum Δ

$$g(\eta_n|\eta_0,\Delta) = \frac{1}{\pi}\frac{\Delta}{(\eta-\eta_0)^2+\Delta^2} \tag{B.2.6}$$

we can carry out the η-integral using the residue theorem and evaluating the function at the pole $\eta_0 - i\Delta$

$$H(z_n;q) = a_q \left[C_0 + \sum_{r=1}^{q} C_r(z_n^r + \bar{z}_n^r) \right] \tag{B.2.7}$$

where $z_n \coloneqq \bar{\alpha}_n(\eta_0 - i\Delta, t)$. In general, the distribution parameters η_0 and Δ can be set independently for each neuron, i.e. distinct η_{0n} and Δ_n. However, in this work, we assume the same Lorentzian distribution for all neurons.

B.3 Coefficient constrains – the dynamical equation

We now turn to the continuum equation of the probability function. Before we insert the Ott/Antonsen ansatz in the continuum equation, the velocity field is best expressed in sinusoidally coupled form as in [LBS13]

$$v = he^{-i\theta_j} + d + \bar{h}e^{i\theta_j} \tag{B.3.1}$$

with

$$h = -\frac{1}{2}(1 - \eta_j - I_j) \qquad \text{and} \qquad d = 1 + \eta_j + I_j \tag{B.3.2}$$

Note that h and d are independent of θ_j. We substitute Eq. (B.3.1) and Eq. (B.1.4) in Eq. (B.1.1), reindex where appropriate, and obtain

$$\left[\dot{\alpha}_j + i\left(\bar{h} + d\alpha_j + h\alpha_j^2\right)\right] \underbrace{\left[\sum_{k=1}^{\infty} k\alpha_j^{k-1}e^{ik\theta_j} + \sum_{k=1}^{\infty} k\bar{\alpha}_j^{k-1}e^{-ik\theta_j} \right]}_{\neq 0} = 0 \tag{B.3.3}$$

Due to the "seemingly miraculous coincidence" [MS09] of having the term in the left bracket as a common factor in the sum, we find a condition for $\alpha_j(\eta_j, t)$. It is a differential equation and constitutes the dynamics of the sub-manifold

$$\dot{\alpha}_j(\eta_j, t) = -i\left(\bar{h} + d\alpha_j(\eta_j, t) + h\alpha_j(\eta_j, t)^2\right) \tag{B.3.4}$$

B.4 Dynamics of the complex order parameter

Typically, one is interested in the mean firing rate or the order parameter itself. Therefore, we compute

$$\int_{-\infty}^{+\infty}\int_0^{2\pi} e^{i\theta_j} f_j(\theta_j, \eta_j, t)d\theta_j d\eta_j = \int_{-\infty}^{+\infty}\int_0^{2\pi} e^{i\theta_j}\frac{g(\eta_n)}{2\pi}\left[1 + \sum_{k=1}^{\infty}\{\alpha_j(\eta_j, t)\}^k e^{ik\theta_j} + c.c.\right]d\theta_j d\eta_j \tag{B.4.1}$$

$$= \int_{-\infty}^{+\infty} g(\eta_n)\bar{\alpha}_j(\eta_j, t)\}d\eta_j \tag{B.4.2}$$

$$= \bar{\alpha}_j(\eta_0 - i\Delta, t) \tag{B.4.3}$$

$$= z_j \tag{B.4.4}$$

Conveniently, we find that the coefficient evaluated at the singularity $\bar{\alpha}_j(\eta_0 - i\Delta)$ actually is the complex order parameter describing the averaged (across the infinite ensemble) dynamics of neuron j. Thus, Eq. (B.3.4) is most relevant for $\eta_j = \eta_0 - i\Delta$ and reads in this case

$$\dot{z}_j = i\left(h + dz_j + \bar{h}z_j^2\right) \tag{B.4.5}$$

$$= \frac{-i(z_j - 1)^2}{2} + \frac{(z_j + 1)^2}{2}\left[-\Delta + i\eta_0 + iJ_j\right] \tag{B.4.6}$$

with

$$J_j = \frac{K}{\langle k\rangle}\sum_{n=1}^{N} A_{jn}H(z_n; q) \tag{B.4.7}$$

The system (B.4.6)-(B.4.7) is a closed dynamical system only depending on all the z_j. Solutions can not leave this $2N$-dimensional sub-manifold, hence it is said to be invariant.

Appendix C

Algorithms

The following algorithms are developed to alter the adjacency matrix, while leaving the degree distribution untouched. Algorithm 1 removes self-edges and Algorithm 2 multi-edges. Algorithm 3 alters an adjacency matrix towards a desired degree assortativity. The four different assortativity coefficients might not be reached without affecting each other. In this case the algorithm has to be applied iteratively to each of the four cases after another until the desired values are reached.

Algorithm 1: Remove self-edges.
Remove self-edges from adjacency matrix A. Pick for each self-edge a random edge from the network and reconnect those two without introducing new multi-edges.

```
1  create list of edges: [e];
2  create list of indices of self-edges in [e]: [s];
3  foreach s in [s] with e_s = (i, i) do
4  |  a_init = A_ii;
5  |  while A_ii = a_init do
6  |  |  pick random index r for [e] with e_r = (k, l);
7  |  |  /* avoid new multi-connections                    */
8  |  |  if A_il = 0 and A_ki = 0 then
9  |  |  |  /* disconnect old edges                        */
10 |  |  |  A_ii = A_ii − 1;
11 |  |  |  A_kl = A_kl − 1;
12 |  |  |  /* connect new edges                           */
13 |  |  |  A_il = A_il + 1;
14 |  |  |  A_ki = A_ki + 1;
15 |  |  |  /* update [e]                                  */
16 |  |  |  e_r = (i, l);
17 |  |  |  e_s = (k, i);
18 |  |  end
19 |  end
20 end
```

Algorithm 2: Remove multi-edges.
Remove multi-edges from adjacency matrix A. Pick for each multi-edge a random edge from the network and reconnect those two without introducing new self- or multi-edges.

```
1  create list of edges: [e];
2  create list of indices of multi-edges in [e]: [m];
3  foreach m in [m] with e_m = (i, j) do
4      /* check if (i,j) is still a multi-edge                    */
5      if A_ij > 1 then
6          a_init = A_ij ;
7          while A_ij = a_init do
8              pick random index r for [e] with e_r = (k, l);
9              /* avoid new self-connections                      */
10             if i ≠ l and j ≠ k then
11                 /* avoid new multi-connections                 */
12                 if A_il = 0 and A_kj = 0 then
13                     /* disconnect old edges                    */
14                     A_ij = A_ij − 1;
15                     A_kl = A_kl − 1;
16                     /* connect new edges                       */
17                     A_il = A_il + 1;
18                     A_kj = A_kj + 1;
19                     /* update [e]                              */
20                     e_r = (i, l);
21                     e_m = (k, j);
22                 end
23             end
24         end
25     end
26 end
```

Algorithm 3: Assortative mixing.
Randomly pair up all N_e edges of the network with adjacency matrix A and reconnect them at once where preferable with respect to target assortativity r_{target}. Repeat the process until the assortativity coefficient lies within the tolerance. Once overshooting the target coefficient, interpolate the length of a shortened list of edge pairs and reconnect those.

```
/* compute difference in assortativity                                  */
Δr = r_target − r(A);
while |Δr| > tolerance do
    pair up all edges [(i, j), (k, l)];
    /* compute whether each pair should be reconnected                  */
    s_Δr = [true: if reconnection will minimise Δr; false: otherwise];
    /* trial reconnection                                               */
    A* = copy(A);
    reconnect edges in A* according to s_Δr;
    Δr* = r_target − r(A*);
    if sign(Δr*) ≠ sign(Δr) then
        /* r(A*) is already beyond the target:                          */
        /* limit number of edges for reconnection process               */
        interpolation data Γ: (0, r(A)), (N_e/2, r(A*));
        while |Δr*| > tolerance do
            interpolate (L, r_target) using Γ;
            s_limit = [true: list index < L; false: list index > L];
            /* trial selection and reconnection                         */
            s* = s_Δr ∧ s_limit;
            A* = copy(A);
            reconnect edges in A* according to s*;
            add (L, r(A*)) to Γ;
            Δr* = r_target − r(A*);
        end
    end
    A = A*;
    Δr = Δr*;
end
```

MASSEY UNIVERSITY
GRADUATE RESEARCH SCHOOL

STATEMENT OF CONTRIBUTION
DOCTORATE WITH PUBLICATIONS/MANUSCRIPTS

We, the candidate and the candidate's Primary Supervisor, certify that all co-authors have consented to their work being included in the thesis and they have accepted the candidate's contribution as indicated below in the *Statement of Originality*.

Name of candidate:	Christian Blasche
Name/title of Primary Supervisor:	Carlo Laing
Name of Research Output and full reference:	
The effects of within-neuron degree correlations in networks of spiking neurons.	
In which Chapter is the Manuscript /Published work:	2
Please indicate:	
• The percentage of the manuscript/Published Work that was contributed by the candidate:	30
and	
• Describe the contribution that the candidate has made to the Manuscript/Published Work:	
The candidate has contributed to the understanding and a numerical implementation of bi-variate correlation based on copulas. He further is responsible for motif detection using open-source tools as well as own algorithms.	
For manuscripts intended for publication please indicate target journal:	
Biological Cybernetics, 2020	
Candidate's Signature:	Christian Bläsche Digitally signed by Christian Bläsche Date: 2020.06.17 09:42:15 +12'00'
Date:	16.06.2020
Primary Supervisor's Signature:	Carlo Laing Digitally signed by Carlo Laing Date: 2020.06.17 10:05:11 +12'00'
Date:	17/6/20

(This form should appear at the end of each thesis chapter/section/appendix submitted as a manuscript/ publication or collected as an appendix at the end of the thesis)

MASSEY UNIVERSITY
GRADUATE RESEARCH SCHOOL

STATEMENT OF CONTRIBUTION
DOCTORATE WITH PUBLICATIONS/MANUSCRIPTS

We, the candidate and the candidate's Primary Supervisor, certify that all co-authors have consented to their work being included in the thesis and they have accepted the candidate's contribution as indicated below in the *Statement of Originality*.

Name of candidate:	Christian Blasche
Name/title of Primary Supervisor:	Carlo Laing
Name of Research Output and full reference:	
Degree assortativity in networks of spiking neurons.	
In which Chapter is the Manuscript /Published work:	3
Please indicate:	
• The percentage of the manuscript/Published Work that was contributed by the candidate:	70
and	
• Describe the contribution that the candidate has made to the Manuscript/Published Work:	
The candidate contributed to the understanding of an assortativity function, the derivative of degree clusters, the generation of all adjacency matrices, the singular value analysis, and computed all results.	
For manuscripts intended for publication please indicate target journal:	
Journal of Computational Dynamics, 2020.	
Candidate's Signature:	Christian Bläsche Digitally signed by Christian Bläsche Date: 2020.06.17 09:41:25 +12'00'
Date:	16.06.2020
Primary Supervisor's Signature:	Carlo Laing Digitally signed by Carlo Laing Date: 2020.06.17 10:06:01 +12'00'
Date:	17/6/20

(This form should appear at the end of each thesis chapter/section/appendix submitted as a manuscript/ publication or collected as an appendix at the end of the thesis)