University of Zurich[UZH]

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Exploring the Parameter Space of Spiking Neural Networks for Winner-Take-All Dynamics

Master's Thesis

Marin Ozaki

Neural Systems and Computation Master's Program
Institute of Neuroinformatics
University of Zurich / ETH Zurich

Supervisors:
Alpha Renner
Dr. Yulia Sandamirskaya

July 15, 2019

Acknowledgements

I would like to express my sincere gratitude to my supervisors, collaborators, and friends that supported me throughout my thesis project and Master's studies.

First and foremost, I would like to thank Alpha Renner and Dr. Yulia Sandamirskaya for their supervision, support, and for allowing me to work on this project in the first place. I further wish to thank Prof. Giacomo Indiveri for helpful feedback, Chenxi for fruitful discussions, and Lukas for proofreading.

I would also like to thank the NSC Master's students and INI members for creating this unique environment that I so greatly enjoyed studying and working in. Special thanks I would like to direct to Lottie Walch for taking care of much of the Master program's administration.

Finally, I wish to thank the *Japan Student Services Organization* for providing funding, *Rice Up!* for providing food, and my family and friends for providing support.

Abstract

Winner-take-all (WTA) networks – circuits composed of recurrently connected populations of excitatory and inhibitory neurons – have been shown to model crucial aspects of cortical processing (Douglas et al., 1989) and provide a powerful framework for a vast range of computations (Maass, 2000). WTA dynamics have been studied extensively in both rate-based and spiking neuron models. Using a strategy similar to the mean-field approach that allows obtaining rate-based population dynamics from spiking neuron equations (Schwalger et al., 2017), we here explore the parameter space of spiking neural networks that results in winner-take-all dynamics with different dynamical properties. In a first step, based on the work by Rutishauser et al. (2011), we derive equations to find parameter ranges for a rate-based neuron model that results in stable soft and hard WTA behavior. We thereby extend their analysis to networks with an arbitrary number of excitatory units. Further, we derive new conditions that separate hysteresis and self-sustained behavior of a WTA network. In a second step, we construct a spiking neural network consisting of groups of excitatory or inhibitory neurons and map its parameters to the rate-based ones. In particular, we map the borders between parameter ranges that separate some of the different dynamical winner-take-all behaviors that we explored in the previous step. Further, we provide a firing rate prediction that proves to be accurate when neurons of the spiking model are weakly connected. For strong connectivity, however, neurons start to synchronize, leading to lower activity of the spiking neuron than predicted, confirming previously discussed limitations of mean-field approaches. The pipeline presented here could prove useful in assisting the tuning of spiking neural network parameters to achieve desired behaviors in the WTA-framework, in particular on neuromorphic hardware.

Contents

List of Figures

Chapter 1

Introduction

1.1 Winner-take-all networks

Winner-take-all (WTA) networks are networks of recurrently connected populations of excitatory and inhibitory neurons that are capable of detecting and amplifying the neural unit receiving the strongest input while suppressing the activity of others (Fang et al., 1996, Feldman and Ballard, 1982). Modeling pivotal aspects of cortical neural networks (Douglas et al., 1989) and providing a powerful computational framework for both software (Maass, 2000) and hardware (Lazzaro et al., 1989), WTA networks have been receiving a vast amount of attention.

In its most typical form, a winner-take-all network consists of several recurrently connected excitatory units as well as a small number of inhibitory units that receive inputs from all excitatory ones and act on them through global inhibition (Rutishauser et al., 2011). WTA networks therefore belong to the computationally powerful class of competitive networks that operate through shared inhibition (Binas et al., 2014, Douglas and Martin, 2007). If, as a result of this competition, the network unit receiving the strongest input is the only one to remain active, this behavior is regarded as hard WTA. Soft WTA, to the contrary, is characterized by the continued activity of multiple units in this scenario.

In contrast to models focusing exclusively on serial feed-forward connections and disregarding the role of excitatory feedback, WTA networks have been put forward as model candidates for canonical microcircuits in the neocortex (Binas et al., 2014). This proposal assumes the existence of a circuit that is repeated, and slightly modified, several times in each area of the cortex (Creutzfeldt, 1977, Douglas et al., 1989, Szentágothai, 1978). Douglas et al. (1989) developed such a simplified circuit model of the visual cortex that was capable of predicting intracellular recordings in the cat striate cortex upon thalamic stimulation. Their model proposes the existence of two excitatory and a single inhibitory interacting neuron population; each receiving input from the thalamus and each other (Douglas et al., 1989).

The self-excitation in this model is in keeping with anatomical studies showing that the vast majority of a neuron's input originates from neighboring excitatory neurons in the same area of the cortex, rather than from long-range projections from other cortical areas or subcortical nuclei (Douglas et al., 1995).

A key question that is discussed in this context is how the relatively small fraction of input that is received from earlier stages of the cortical hierarchy, compared to the predominant self-excitation, can be processed reliably (Douglas and Martin, 2007). To investigate the computational significance of this circuitry, Douglas and Martin (2007) analyzed a simple network of linear threshold neurons that, coarsely inspired by the recurrent circuitry of the neocortex, comprised a single inhibitory neuron as well as a large population of excitatory neurons that receive input through excitatory feed-forward and feedback connections as well as inhibitory feedback from a global inhibitory neuron. Their simulations revealed that the excitatory recurrent feedback results in targeted enhancement of input features that are aligned with the patterns of the feedback connection weights. The global inhibitory neuron further imposes a dynamical inhibitory threshold to suppress outliers (Douglas and Martin, 2007).

Studies like this have been used to demonstrate that non-linear functions such as signal amplification and restoration can be implemented through simple WTA circuits. Simulational studies have further been complemented by more theoretical analyses. Maass (2000), for instance, showed that the winner-take-all computation is remarkably powerful compared to computation in threshold and sigmoidal gates and that circuits making use of a single soft-WTA gate can approximate an arbitrary continuous function. The underlying winner-take-all computation has further been put forward as a model of processing in the cortex. Among the most frequently referenced examples are the vision model developed by Riesenhuber and Poggio (1999) and the attention model by Itti et al. (1998).

1.2 Different implementations of WTA networks

Winner-take-all networks can be implemented in both rate-based and spiking neural networks. While spiking models describe a neuron's output in the form of discrete spikes and thereby allow the use of time for carrying out computation (Maass, 1997), rate-based models, due to their simplicity, allow for the construction of large-scale networks and mathematical analysis of their behavior. As both types of models are used throughout this thesis, I will briefly introduce and contextualize them here and provide an in-depth formalization in the second and third chapter.

1.2.1 Rate-based neuron models

Rate-based neuron models are built on the idea that a substantial fraction of what a neuron encodes can already be captured by its average firing rate. While it has been proposed that stimulus information are encoded by the relative timing of individual spikes in some systems (Montemurro et al., 2008, Shapiro and Ferbinteanu, 2006); in other instances, the firing frequency was shown to already convey much information. The latter view thereby dates back to the work of Adrian and Zotterman (1926) who demonstrated the proportionality of firing frequency and stimulus intensity.

The simplicity and computational tractability of rate-based models allow for the construction of large-scale networks and detailed mathematical analysis. A powerful concept in this context is that of neural fields, where the continuous spatiotemporal evolution of quantities in those fields, such as average firing rates, can be modeled through a set of neural field equations (Coombes, 2006).

Beurle (1956) is believed to be the first to have approached approximating such a spatial continuum of neural activity for a network of exclusively excitatory neurons. Wilson and Cowan (1972, 1973) extended their work to model refractory periods and allow for the incorporation of inhibitory neurons into the networks. At the core of their model are differential equations describing the temporal evolution of the average activity of neuron populations (Wilson and Cowan, 1973). In order to study the behavior of a large population of neurons, they thereby employed a mean-field approach. Amari (1977) followed up on this work, introducing local excitation and distal inhibition, a powerful model for interacting populations of excitatory and inhibitory neurons. This work still forms the mathematical foundation for dynamic neural fields, the computational building blocks for dynamic field theory – a framework to describe elementary cognitive functions as a consequence of neuronal population dynamics (Schöner, 2008).

To sum, rate-based models and their related computational and conceptual frameworks facilitate large-scale modeling of spatiotemporally continuous neural fields and mathematical analysis of their behavior.

1.2.2 Spiking neuron models

Spiking neuron models are the second class of models that are dealt with in this thesis. These biologically more plausible models of neural function have been proposed to form the third generation of neural network models (Maass, 1997). They thereby follow the first generation of McCullough-Pitts neurons (McCulloch and Pitts, 1943) and the second generation of units that applied an activation function with continuous output to a weighted input sum (Maass, 1997).

Within the class of spiking networks, the many neuron models that have

been established primarily differ in the degree to which they abstract biological detail (Herz et al., 2006). Among the complex neuron models are, for instance, the model of Hodgkin and Huxley (1952), which describe biophysical processes at the level of individual ion channels, as well as compartmental models (Segev et al., 1989) that also account for the spatial structure of a neuron.

While more abstract models mimic biophysical processes less closely, they allow for a more systematic analysis of their key computing elements as well as the simulation of networks incorporating a large number of units. A popular choice among simplified spiking neuron models is the leaky-integrate-and-fire (LIF) neuron that accumulates the input and generates a spike when exceeding a certain threshold (Gerstner and Kistler, 2002). The key computation in this model is the temporal summation of inputs. In this thesis, leaky-integrate-and-fire units will be utilized as computational building blocks for spiking neural networks.

To sum, though representing one of the simplest instances of spiking neuron models, compared to rate-based models, a network of LIF neurons more realistically models a biological neuron's output – a discrete spike – and thereby allows making use of temporal information in its computations (Maass, 1997).

1.3 Goals and structure of this thesis

The two neuron models presented have both been used extensively for the study of WTA dynamics. Using a strategy similar to the mean-field approach that allows to obtain rate-based population dynamics from spiking neuron equations (Schwalger et al., 2017), in this thesis, I aim to explore the parameter space of spiking neural networks that results in winner-take-all dynamics with different dynamical properties. This effort can be divided into two main parts.

First, based on Rutishauser et al. (2011), I will derive equations to find parameter ranges for a rate-based neuron model that results in stable soft and hard WTA behavior. I will thereby extend the presented approach to allow stability analysis in networks with an arbitrary number of excitatory units. Furthermore, I will derive new conditions that separate hysteresis and self-sustained behavior of the network. This work is described in the second chapter.

Second, using a strategy resembling the mean-field approach (Gerstner, 2000), I will construct a spiking neural network consisting of groups of excitatory or inhibitory neurons and map its parameters to the rate-based ones. In particular, I will map the borders between parameter ranges that separate some of the winner-take-all behaviors that I explored in the previous step. This work is described in the third chapter.

Chapter 2

Continuous model dynamics

In this chapter, we will focus on the analysis of rate-based models with regard to winner-take-all dynamics. Special emphasis will thereby be placed on the work of Rutishauser et al. (2011) who propose a formalism for neuronal activity within simple WTA networks and present an analytical approach for assessing their stability. Here, we start by introducing the model and reproduce, and slightly modify, their stability analysis. Next, we extend the analyses that were originally carried out on a three-unit WTA network to networks with an arbitrary number of excitatory units and a single inhibitory unit. In addition to stability conditions in hard and soft WTA regimes, we will derive conditions for two other phenomena: hysteresis and self-sustained behavior. Those will be introduced in later parts of this chapter.

2.1 The WTA model by Rutishauser et al.

The basic architecture of a winner-take-all (WTA) network is illustrated in Figure 2.1, adapted from Rutishauser et al. (2011). A WTA network, here, comprises N units, out of which N-1 are excitatory and a single one is inhibitory. Each excitatory unit $(u_1, ..., u_{N-1})$ thereby receives input from both its neighbors (α_2) and itself (α_1); the inhibitory unit receives input from each excitatory unit (β_1) and inhibits each excitatory unit (β_2). Network A (see Figure 2.1) is restricting an excitatory neuron's input to self-excitation (α_1), while network A' (see Figure 2.1) also takes input from its neighbors into account. The dynamics of these units, as described in Rutishauser et al. (2011), are given in Eqs. 2.1-2.3, where $u_1, ..., u_{N-1}$ indicate the activity of the excitatory units and u_N corresponds to that of the inhibitory unit.

$$\tau \frac{du_i}{dt} = -gu_i + F(\alpha_1 u_i - \beta_2 u_N + I_i - T_i) \quad (2.1)$$

$$\tau \frac{du_N}{dt} = -gu_N + F(\beta_1 \sum_{j=1}^{N-1} u_j - T_N) \quad (2.2)$$

$$F(x) = max(0, x) \quad (2.3)$$

Here, I_i represents the external input to a unit i, g stands for the conductance, and T_i indicates the threshold for each unit, which is constant and identical across all units. The conductance and thresholds as well the parameters α_1, β_1 and β_2 are, by definition, required to be greater than 0.

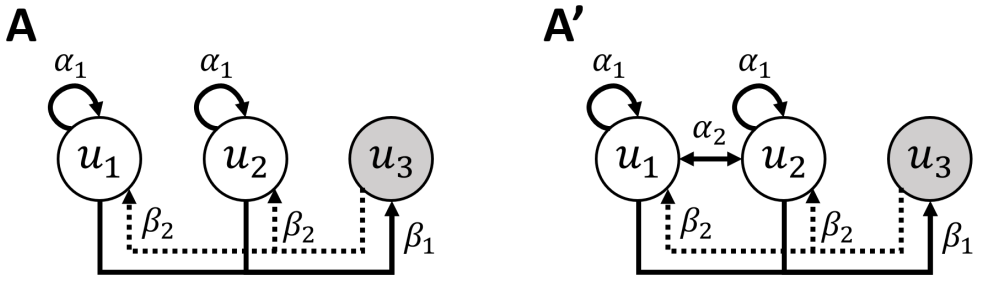


Figure 2.1: Basic architecture of a 3-unit winner-take-all network, modified from Rutishauser et al. (2011). Here, u_1 and u_2 represent excitatory units and u_3 a single inhibitory unit. Each excitatory unit receives input from both its neighbors (α_2) and itself (α_1); the inhibitory unit receives input from each excitatory unit (β_1) and sends back inhibition (β_2). α_2 is disregarded in network A but taken into account in network A'. Analytical results for both network types are provided in Rutishauser et al. (2011).

2.2 Stability analysis

Inspired by the analysis carried out in Rutishauser et al. (2011), we engaged in two complementary approaches to assess the stability of WTA networks: first, we made use of the Jacobian; second, of the Hermitian.

2.2.1 Jacobian analysis

The Jacobian $\mathbf{J_A}$ for the network A in Fig. 2.1, governed by Eqs. 2.1-2.3, is given by:

$$\tau \mathbf{J_A} = \begin{bmatrix} l_1\alpha_1 - g & 0 & -l_3\beta_2 \\ 0 & l_2\alpha_1 - g & -l_3\beta_2 \\ l_1\beta_1 & l_2\beta_1 & -g \end{bmatrix} \tag{2.4}$$

Here, l_k is a dummy variable that takes the value of either 0 or 1, based on the derivative of $F(x) = \max(0, x)$. By setting only a single l_k to 1 and all others to 0, we can describe hard WTA (with l_k being the winning unit). Setting all l_k to 1 implements soft WTA. In both cases, by definition, the following set of conditions has to hold: $g > 0$, $\alpha_1 > 0$, $\beta_1 > 0$, and $\beta_2 > 0$.

Conditions for hard WTA

We begin with the derivation of stability conditions for hard WTA networks. Accordingly, we set $l_1 = 1$, $l_2 = 0$, and $l_3 = 1$, with u_1 being the winner. The Jacobian $\mathbf{J_{Ah}}$ for the hard-WTA configuration of network A in Fig. 2.1, governed by Eqs. 2.1-2.3, is given by:

$$\tau \mathbf{J_{Ah}} = \begin{bmatrix} \alpha_1 - g & 0 & -\beta_2 \\ 0 & -g & -\beta_2 \\ \beta_1 & 0 & -g \end{bmatrix} \tag{2.5}$$

Our goal is to assess the stability of this system. We know from contraction analysis that $\mathbf{J_{Ah}}$ has to be negative definite in order for the system to be stable (for a detailed derivation, see Izhikevich (2007)). This means that all real parts of the eigenvalues λ of $\mathbf{J_{Ah}}$ have to be negative. By solving $\det(\lambda\mathbf{I} - \mathbf{J_{Ah}}) = 0$, we get the eigenvalues:

$$\lambda = \begin{pmatrix} -g \\ \frac{\alpha_1}{2} - \frac{\sqrt{\alpha_1^2 - 4\,\beta_1\,\beta_2}}{2} - g \\ \frac{\alpha_1}{2} + \frac{\sqrt{\alpha_1^2 - 4\,\beta_1\,\beta_2}}{2} - g \end{pmatrix} \tag{2.6}$$

Next, we can check weather $\mathbf{Re}(\lambda) < 0$ holds for each of the eigenvalues. We see that the first eigenvalue always satisfies the condition ($-g < 0$). For the second and third eigenvalue, the real part depends on the sign of $\alpha_1^2 - 4\beta_1\beta_2$. We can carry out a case differentiation:

Case I: When $\alpha_1^2 - 4\beta_1\beta_2 < 0$:

$$\mathbf{Re}(\lambda) \quad = \quad \begin{pmatrix} -g \\ \frac{a_1}{2} - g \\ \frac{a_1}{2} - g \end{pmatrix} \tag{2.7}$$

Setting $\mathbf{Re}(\lambda) < 0$, this condition is fulfilled for all eigenvalues when:

$$a_1 < 2g \tag{2.8}$$

Case II: When $\alpha_1^2 - 4\beta_1\beta_2 \geq 0$:

$$\mathbf{Re}(\lambda) \quad = \quad \begin{pmatrix} -g \\ \frac{\alpha_1}{2} - \frac{\sqrt{\alpha^2 - 4\,\beta_1\,\beta_2}}{2} - g \\ \frac{\alpha_1}{2} + \frac{\sqrt{\alpha^2 - 4\,\beta_1\,\beta_2}}{2} - g \end{pmatrix} \tag{2.9}$$

Setting $\mathbf{Re}(\lambda) < 0$, this condition is fulfilled for all eigenvalues when the following two conditions hold:

$$(i)\ a_1 - 2g \quad < \quad \sqrt{\alpha^2 - 4\,\beta_1\,\beta_2} \tag{2.10}$$

$$(ii)\ \sqrt{\alpha^2 - 4\,\beta_1\,\beta_2} \quad < \quad 2g - a_1 \tag{2.11}$$

Given $0 \leq \sqrt{\alpha^2 - 4\,\beta_1\,\beta_2}$ (see case II condition), we can rewrite Eq. 2.11 as:

$$(ii.1)\ \alpha_1 \quad < \quad 2g \tag{2.12}$$

$$(ii.2)\ \beta_1\beta_2 \quad > \quad (\alpha_1 - g)g \tag{2.13}$$

Given Eq. 2.12, Eq. 2.10 is always true. The conditions for the second case can therefore be reduced to Eqs. 2.12 and 2.13. The condition for the first case is given by Eq. 2.8. Merging the results from this case differentiation, Fig. 2.2 depicts the parameter ranges that lead to stable hard WTA behavior.

Conditions for soft WTA

We can reuse the same pipeline to derive the conditions for soft WTA by simply modifying our dummy variables: $l_1 = l_2 = l_3 = 1$. The Jacobian $\mathbf{J_{As}}$ for the soft-WTA configuration of network A in Fig. 2.1, governed by Eqs. 2.1-2.3, is then provided by:

$$\tau\mathbf{J_{As}} = \begin{bmatrix} \alpha_1 - g & 0 & -\beta_2 \\ 0 & \alpha_1 - g & -\beta_2 \\ \beta_1 & \beta_1 & -g \end{bmatrix} \tag{2.14}$$

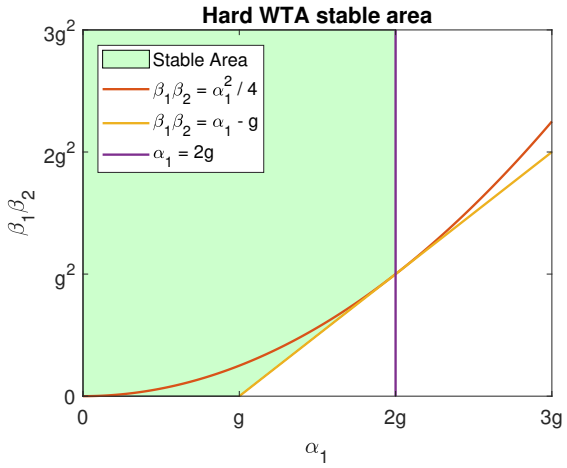


Figure 2.2: Depiction of parameter ranges for stable hard-WTA behavior in the 3-unit network type A depicted in Fig. 2.1. Parameters combinations chosen from the green area result in stable behavior. Note that the units for α_1 are given in g while the units for $\beta_1\beta_2$ are given in g^2.

To assess stability, we again determine the conditions that need to hold in order for the real part of all eigenvalues of the Jacobian to be negative. By solving $\det(\lambda\mathbf{I} - \mathbf{J_{As}}) = 0$, we can extract the eigenvalues λ:

$$\lambda = \begin{pmatrix} \alpha_1 - g \\ \frac{\alpha_1}{2} - \frac{\sqrt{\alpha_1^2 - 8\,\beta_1\,\beta_2}}{2} - g \\ \frac{\alpha_1}{2} + \frac{\sqrt{\alpha_1^2 - 8\,\beta_1\,\beta_2}}{2} - g \end{pmatrix} \tag{2.15}$$

By solving $\mathbf{Re}(\lambda) < 0$, we get the conditions for soft WTA:

Case I: When $\alpha^2 - 8\beta_1\beta_2 < 0$:

$$\alpha_1 < g \tag{2.16}$$

Case II: When $\alpha^2 - 8\beta_1\beta_2 \geq 0$:

$$\alpha_1 < g \tag{2.17}$$

$$\beta_1\beta_2 > \frac{(\alpha_1 - g)g}{2} \tag{2.18}$$

From Eq. 2.17 and the parameter definition ($\beta_1 > 0$ and $\beta_2 > 0$), it follows that Eq. 2.18 is always true. The condition for the second case can therefore be reduced to Eq. 2.17: $\alpha_1 < g$. The condition for the first case is given by Eq. 2.16. Merging both, Fig. 2.3 provides an overview of the stable area for both soft and hard WTA.

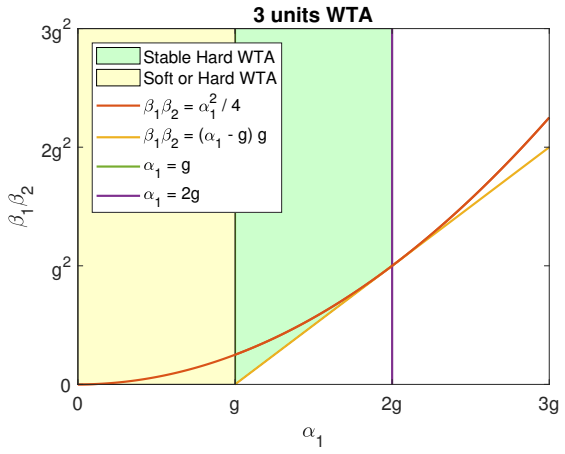


Figure 2.3: Depiction of parameter ranges for stable hard and soft WTA behavior in the 3-unit network type A depicted in Fig. 2.1. Parameter sets chosen from the green area result in stable hard WTA behavior; those chosen from the yellow area in soft or hard WTA behavior. Note that the units for α_1 are given in g while the units for $\beta_1\beta_2$ are given in g^2.

2.2.2 Hermitian analysis

A second approach to derive analytical conditions for network stability, as detailed in Rutishauser et al. (2011), can be described as follows:

1) Carry out an eigendecomposition of the Jacobian $\mathbf{J} = \mathbf{Q\Lambda Q^{-1}}$
2) Set $\mathbf{\Theta} = \mathbf{Q}^{-1}$ and $\mathbf{F} = \mathbf{\Theta J \Theta^{-1}}$
3) Get the Hermitian part of $\mathbf{F}$, $\mathbf{F_H} = \frac{1}{2}(\mathbf{F} + \mathbf{F}*^T)$
4) Check whether $\mathbf{F_H}$ is negative definite.

Conditions for hard WTA based on Hermitian

After carrying out the steps detailed above, the eigenvalues λ of $\mathbf{F_H}$ for $\mathbf{J_{Ah}}$ are given by:

$$\lambda = \begin{pmatrix} -g \\ \frac{\alpha_1}{2} + \frac{\left(\alpha_1^2 - 4\,\beta_1\,\beta_2\right)^{3/2}}{4\left|\alpha_1^2 - 4\,\beta_1\,\beta_2\right|} + \frac{\sqrt{\alpha_1^2 - 4\,\beta_1\,\beta_2}}{4} - g \\ \frac{\alpha_1}{2} - \frac{\left(\alpha_1^2 - 4\,\beta_1\,\beta_2\right)^{3/2}}{4\left|\alpha_1^2 - 4\,\beta_1\,\beta_2\right|} - \frac{\sqrt{\alpha_1^2 - 4\,\beta_1\,\beta_2}}{4} - g \end{pmatrix}, \tag{2.19}$$

where $\alpha_1^2 - 4\beta_1\beta_2 \neq 0$. To guarantee network stability, the real parts of these eigenvalue have to be negative. By definition of g, the first eigenvalue is always

smaller than zero. The sign of the second and third eigenvalue depend on the sign of $\alpha_1^2 - 4\beta_1\beta_2$. We therefore carry out a case differentiation and, in each case, solve for negative eigenvalues. The results are summarized below:

Case I: When $\alpha_1^2 - 4\beta_1\beta_2 < 0$:

$$\alpha_1 < 2g \tag{2.20}$$

Case II: When $\alpha_1^2 - 4\beta_1\beta_2 > 0$:

$$\lambda = \begin{pmatrix} -g \\ \frac{\alpha_1}{2} - \frac{\sqrt{\alpha^2 - 4\,\beta_1\,\beta_2}}{2} - g \\ \frac{\alpha_1}{2} + \frac{\sqrt{\alpha^2 - 4\,\beta_1\,\beta_2}}{2} - g \end{pmatrix} \tag{2.21}$$

Setting $\mathbf{Re}(\lambda) < 0$, we get as conditions:

$$\alpha_1 < 2g \tag{2.22}$$

$$\beta_1\beta_2 > (\alpha_1 - g)g \tag{2.23}$$

The conditions derived using the Hermitian approach are identical to those derived using the Jacobian approach. This has also been the case for soft WTA (results not shown here). In the discussion of their paper, however, Rutishauser et al. (2011) conclude that they did not succeed in deriving analytical conditions using the Jacobian.

Furthermore, they derived stronger conditions as they appear to be neglecting the parameter region where $\alpha_1^2 - 4\beta_1\beta_2 < 0$. As a consequence, they note that their analytical solution assigns an upper bound to the parameter β_2 which is, in fact, not necessary with regard to their simulations. Using our approach, by carrying out a full case differentiation, no such upper bound is set and analytical and simulation results are identical. Apart from this, our results match those reported by Rutishauser et al. (2011). We will use the Jacobian method hereafter.

2.2.3 Numerical simulations

To validate the analytical results, we carried out numerical simulations of the differential equations of the model by Rutishauser et al. (2011) for 3-unit WTA networks (Eqs. 2.1 - 2.3). For these simulations, we made use of the simulator Brian 2 (Goodman and Brette, 2008).

Fig. 2.4 shows the stable and unstable behaviour in the simulation, the borders of which match well with the analytical result. The unstable behavior, observed outside of the stable soft or hard WTA parameter region, can further

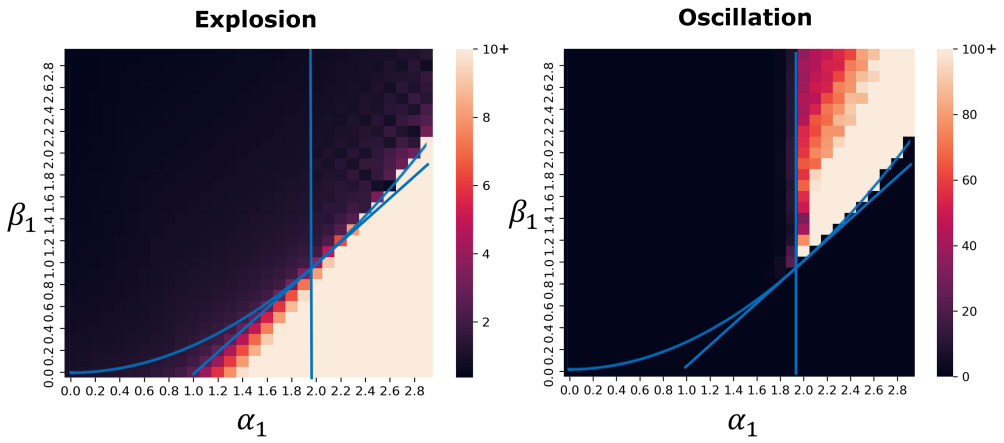


Figure 2.4: Results of numerical simulations of the model by Rutishauser et al. (2011) with regard to stability. Numerical simulations were conducted for Eqs. 2.1 - 2.3 governing the activity of the 3-unit network. For all simulations, the parameters β_2 and g were set to 1, and β_1 and α_1 were varied between 0 and 2.8 by steps of 0.1. For each parameter set, a simulation was carried out for 2 seconds, where from 0.5 to 1.5 seconds, the winning unit received an external input of 10 and the losing unit an input of 8. In the other time ranges, no external input was provided. Left panel: Visualization of explosion in the parameter space. We measured the average activity for the winning unit u_1 between 1.4 and 1.5 seconds, i.e., the last 100 ms of the input period, and plotted these values as a heatmap. To facilitate visual inspection of such heatmap, the mean of u_1 was logarized with the base of 10 and all values exceeding 10 are depicted by the same maximum color of the color bar. The blue lines represent the borders for the Jacobian analysis as shown in Fig. 2.3. When $\alpha_1 < 2$, the explosion area is given by $\beta_1\beta_2 < \alpha_1 - 1$. When $\alpha_1 > 2$, $\beta_1\beta_2 < \alpha_1^2/4$ make up the explosion area. This result matches with the simulation presented in Rutishauser et al. (2011). Right panel: Visualization of oscillation in the parameter space. To quantify oscillation, we measured the sum of the negative gradient: if v denotes the activity vector and grad[t]=v[t]-v[t-1] its gradient, all negative values in the gradient were summed up here. To differentiate between explosion and oscillation, only negative values were summed up. As we can see in the heatmap, in the area where $\beta_1\beta_2 < \alpha_1^2/4$ holds, we can find oscillating behavior.

be subdivided into two classes: first, explosion, where the activity level goes up to infinity; second, oscillation, where an excitatory and inhibitory unit oscillate. The details are provided in Fig. 2.4.

Fig. 2.5 shows the result of numerical simulations used to assess soft and hard WTA. To measure soft and hard WTA, we calculated the average activity for the winner unit, u_1, in the last 100 ms of the input period (1 second in total) and

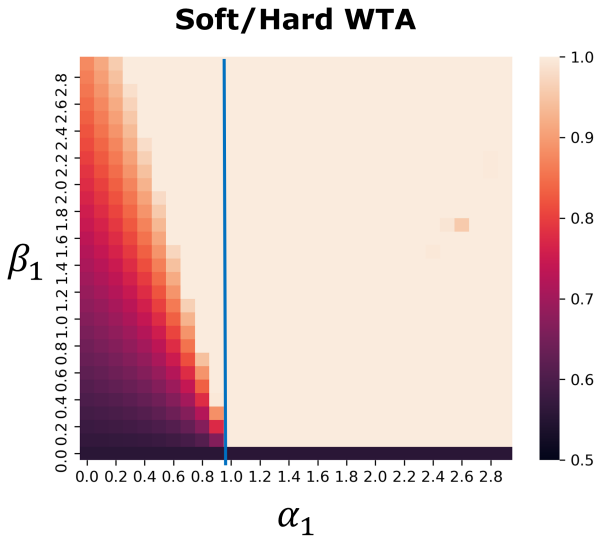


Figure 2.5: Results of numerical simulations of the model by Rutishauser et al. (2011) with regard to soft and hard WTA. The simulation configuration used for generating this figure is the same as the one used for Fig. 2.4. The colors indicate the relative fraction of the winning unit's activity. The blue line depicts the theoretical soft/hard WTA boundary as shown in Fig. 2.3.

determined the ratio that the summed up activity of this unit would make up out of the overall activity of all units. This value was plotted in the heatmap shown in Fig. 2.5. In the absence of any interaction, this value would correspond to $\frac{10}{10+8} = 0.556$, whereas in hard WTA, it would be 1. The blue line depicts the theoretical soft/hard WTA boundary as it appears in Fig. 2.3. We find that the entire parameter space right of the blue line ($\alpha_1 > 1$) does, indeed, hold values close to 1 and corresponds to hard WTA. We further note that in the area left of the blue line ($\alpha_1 < 1$), both soft and hard WTA can be stable. Further, it is shown that within this area, when self-excitation and inhibition are large, the WTA configuration will be hard; otherwise soft. Overall, this investigation validates the analytically obtained conditions.

2.3 Extensions of the model

2.3.1 3 units = (2 Exc, 1 Inh), with excitation (α_1, α_2)

Here, we start to extend the network and carry out the Jacobian analysis detailed in the previous section. As a first step, we add inter-excitatory interactions, thereby producing the network type depicted in Fig. 2.1 A'. The Jacobian $\mathbf{J_{A'}}$ for such network is given by:

$$\tau \mathbf{J}_{\mathbf{A}'} = \begin{bmatrix} l_1\alpha_1 - g & l_2\alpha_2 & -l_3\beta_2 \\ l_1\alpha_2 & l_2\alpha_1 - g & -l_3\beta_2 \\ l_1\beta_1 & l_2\beta_1 & -g \end{bmatrix}, \tag{2.24}$$

where $g > 0$, $\alpha_1 > 0$, $\alpha_2 > 0$, $\beta_2 > 0$, and $\beta_1 > 0$ hold by definition.

Conditions for hard WTA

Setting $l_1 = 1$, $l_2 = 0$, $l_3 = 1$ and repeating the Jacobian analysis presented above, we can derive conditions for stability in the hard WTA regime. These conditions are identical to those of the 3-unit WTA without inter-excitatory unit interactions:

$$\alpha_1 < 2g \tag{2.25}$$
$$\beta_1\beta_2 > (\alpha_1 - g)g \tag{2.26}$$

This result is intuitively plausible as there is no additional interaction with the winner unit when all other excitatory units are inactive.

Conditions for soft WTA

Setting $l_1 = l_2 = l_3 = 1$, we can carry out the same analysis for soft WTA in the network shown in Fig. 2.1 A'. The Jacobian is given by Eq. 2.27 and its eigenvalues by Eq. 2.28:

$$\tau \mathbf{J}_{\mathbf{A's}} = \begin{bmatrix} \alpha_1 - g & \alpha_2 & -\beta_2 \\ \alpha_2 & \alpha_1 - g & -\beta_2 \\ \beta_1 & \beta_1 & -g \end{bmatrix} \tag{2.27}$$

$$\lambda = \begin{pmatrix} \alpha_1 - \alpha_2 - g \\ \frac{\alpha_1}{2} + \frac{\alpha_2}{2} - \frac{\sqrt{{\alpha_1}^2 + 2\,\alpha_1\,\alpha_2 + {\alpha_2}^2 - 8\,\beta_1\,\beta_2}}{2} - g \\ \frac{\alpha_1}{2} + \frac{\alpha_2}{2} + \frac{\sqrt{{\alpha_1}^2 + 2\,\alpha_1\,\alpha_2 + {\alpha_2}^2 - 8\,\beta_1\,\beta_2}}{2} - g \end{pmatrix} \tag{2.28}$$

By substituting $\alpha = \alpha_1 + \alpha_2$, the eigenvalues can be expressed as:

$$\lambda = \begin{pmatrix} \alpha_1 - \alpha_2 - g \\ \frac{\alpha}{2} - \frac{\sqrt{\alpha^2 - 8\,\beta_1\,\beta_2}}{2} - g \\ \frac{\alpha}{2} + \frac{\sqrt{\alpha^2 - 8\,\beta_1\,\beta_2}}{2} - g \end{pmatrix} \tag{2.29}$$

Here, we carry out a case differentiation and derive the conditions that need to hold to ensure negative real parts of all eigenvalues.

Case I: When $\alpha^2 - 8\beta_1\beta_2 < 0$:

$$\begin{aligned} \alpha_1 - \alpha_2 &< g && (2.30)\\ \alpha &< 2g && (2.31) \end{aligned}$$

Case II: When $\alpha^2 - 8\beta_1\beta_2 \geq 0$:

$$\begin{aligned} \alpha_1 - \alpha_2 &< g && (2.32)\\ \alpha &< 2g && (2.33)\\ \beta_1\beta_2 &> \frac{(\alpha - g)g}{2} && (2.34) \end{aligned}$$

In addition, in both cases, $\alpha_1 < \frac{3}{2}g$ and $\alpha_2 < \frac{1}{2}g$ hold because:

$$\begin{aligned} \alpha_1 - \alpha_2 &< g && (2.35)\\ \alpha_1 + \alpha_2 &< 2g && (2.36) \end{aligned}$$

Adding both equations yields:

$$\begin{aligned} 2\alpha_1 &< 3g && (2.37)\\ \alpha_1 &< \frac{3}{2}g && (2.38) \end{aligned}$$

By subtracting Eq. 2.38 from Eq. 2.36, we get:

$$\alpha_2 < \frac{1}{2}g \tag{2.39}$$

Fig. 2.6 shows the resulting parameter range for stable soft WTA behavior. Note that an additional condition is given by $\alpha_1 < \frac{3}{2}g$. Stability in hard WTA is not depicted here as the conditions are identical to those shown in Fig. 2.2.

2.3.2 4 units = (3 Exc, 1 Inh), with excitation (α_1, α_2)

Fig. 2.7 shows two architectures, B and B', for a 4-unit WTA network. Eq. 2.40 provides its Jacobian. We will here demonstrate the analysis for network type B'.

$$\mathbf{J_{B'}} = \begin{pmatrix} l_1\alpha_1 - g & l_2\alpha_2 & l_3\alpha_2 & -l_4\beta_2 \\ l_1\alpha_2 & l_2\alpha_1 - g & l_3\alpha_2 & -l_4\beta_2 \\ l_1\alpha_2 & l_2\alpha_2 & l_3\alpha_1 - g & -l_4\beta_2 \\ l_1\beta_1 & l_2\beta_1 & l_3\beta_1 & -g \end{pmatrix} \tag{2.40}$$

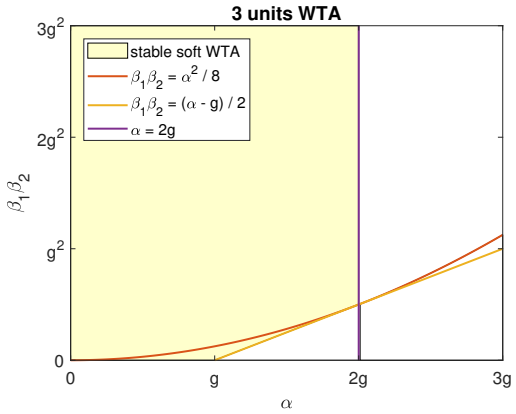


Figure 2.6: Depiction of parameter ranges for stable soft WTA behavior in 3-unit networks with inter-excitatory connections. Parameter sets chosen from the yellow area result in stable soft WTA behavior as long as $\alpha_1 - \alpha_2 < 0$ holds. The horizontal axis α represents the sum of α_1 and α_2. Note that the units for α are given in g while the units for $\beta_1\beta_2$ are given in g^2.

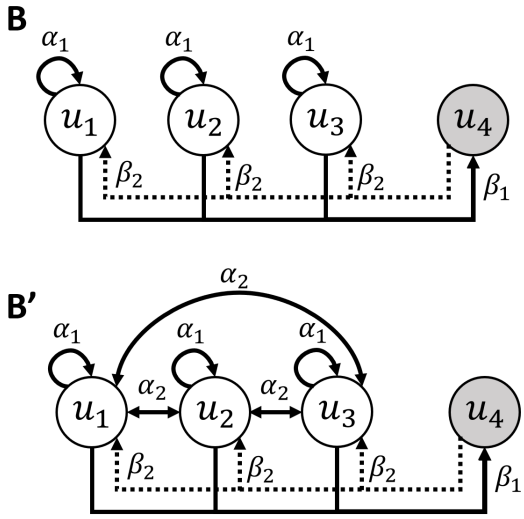


Figure 2.7: Basic architecture of 4-unit WTA networks, where u_1, u_2, and u_3 represent excitatory units and u_4 represents a single inhibitory unit. Each excitatory unit receives input from both its neighbors (α_2) and itself (α_1); the inhibitory unit receives input from each excitatory unit (β_1) and sends back inhibition (β_1). Note that the inter-excitatory connection α_2 is disregarded in network B but taken into account in network B'.

Note that the stability conditions for hard WTA are identical for different numbers of excitatory units as, in the end, only the winning unit remains active. Therefore, we will, in this and the following derivation, focus exclusively on the

soft WTA stability conditions. The Jacobian for the soft WTA configuration can be obtained by replacing all activation variables l_k in Eq. 2.40 by 1:

$$\mathbf{J_{B's}} = \begin{pmatrix} \alpha_1 - g & \alpha_2 & \alpha_2 & -\beta_2 \\ \alpha_2 & \alpha_1 - g & \alpha_2 & -\beta_2 \\ \alpha_2 & \alpha_2 & \alpha_1 - g & -\beta_2 \\ \beta_1 & \beta_1 & \beta_1 & -g \end{pmatrix} \tag{2.41}$$

Calculating the eigenvalues of this Jacobian, we get:

$$\lambda = \begin{pmatrix} \alpha_1 - \alpha_2 - g \\ \alpha_1 - \alpha_2 - g \\ \frac{\alpha_1}{2} + \alpha_2 + \frac{\sqrt{{\alpha_1}^2 + 4\,\alpha_1\,\alpha_2 + 4{\alpha_2}^2 - 12\,\beta_2\,\beta_1}}{2} - g \\ \frac{\alpha_1}{2} + \alpha_2 - \frac{\sqrt{{\alpha_1}^2 + 4\,\alpha_1\,\alpha_2 + 4{\alpha_2}^2 - 12\,\beta_2\,\beta_1}}{2} - g \end{pmatrix}, \tag{2.42}$$

where $\mathrm{i} = \sqrt{-1}$. By substituting $\alpha = \alpha_1 + 2\alpha_2$, the eigenvalues can be expressed as:

$$\lambda = \begin{pmatrix} \alpha_1 - \alpha_2 - g \\ \alpha_1 - \alpha_2 - g \\ \frac{\alpha}{2} - \frac{\sqrt{\alpha^2 - 12\,\beta_1\,\beta_2}}{2} - g \\ \frac{\alpha}{2} + \frac{\sqrt{\alpha^2 - 12\,\beta_1\,\beta_2}}{2} - g \end{pmatrix} \tag{2.43}$$

From the identical first two eigenvalues, it follows that $\alpha_1 - \alpha_2 < g$ must hold. For assessing the last two eigenvalues, we engage in a case differentiation.

Case I: $\alpha^2 - 12\beta_1\beta_2 < 0$:

$$\alpha_1 - \alpha_2 < g \tag{2.44}$$
$$\alpha < 2g \tag{2.45}$$

Case II: $\alpha^2 - 12\beta_1\beta_2 \geq 0$:

$$\alpha_1 - \alpha_2 < g \tag{2.46}$$
$$\alpha < 2g \tag{2.47}$$
$$\beta_1\beta_2 > \frac{1}{3}(\alpha - g)g \tag{2.48}$$

Further, in both cases it holds that:

$$\alpha_1 < \frac{4}{3}g \tag{2.49}$$

$$\alpha_2 < \frac{1}{3}g \tag{2.50}$$

These are the conditions for soft WTA stability in a 4-unit network. They qualitatively resemble those obtained for the 3-unit network and are therefore no longer depicted. Conditions for network B in Fig. 2.7 can be obtained by setting $\alpha_2 = 0$ in the conditions for network B' above.

2.3.3 n+1 units = (n Exc, 1 Inh), with excitation (α_1, α_2)

So far, we derived conditions for the stability of soft and hard WTA networks with a fixed number of units. Here, we wish to generalize some of these results to networks with n excitatory and a single inhibitory unit, connected as shown in the 4-unit instance depicted in Fig. 2.7 B'. Carefully inspecting the results presented in the previous sections, one could hypothesize that two eigenvalues of the network's Jacobian are given by:

$$\frac{\alpha}{2} \pm \frac{\sqrt{\alpha^2 - 4n'\,\beta_1\,\beta_2}}{2} - g, \tag{2.51}$$

where α is the sum of all inputs and n' the number of active excitatory units. The remaining $n-2$ eigenvalues seems to define the relationships between the strength of self-excitation and other forms of excitation.

In the following pages, through the means of mathematical induction, we wish to prove that the eigenvalues λ of the Jacobian of our network, given in Eq. 2.52, are those provided in Eq. 2.53. We will thereby assume that all connections between excitatory unit pairs is given by the same weight (α_2).

$$\mathbf{J_{ns}} = \begin{pmatrix} \alpha_1 - g & \alpha_2 & \cdots & \alpha_2 & -\beta_2 \\ \alpha_2 & \alpha_1 - g & \cdots & \alpha_2 & -\beta_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_2 & \alpha_2 & \cdots & \alpha_1 - g & -\beta_2 \\ \beta_1 & \beta_1 & \cdots & \beta_1 & -g \end{pmatrix} \tag{2.52}$$

$$\lambda = \begin{pmatrix} \left.\begin{matrix} \alpha_1 - \alpha_2 - g \\ \alpha_1 - \alpha_2 - g \\ \vdots \\ \alpha_1 - \alpha_2 - g \end{matrix}\right\} n-1 \\ \\ \frac{\alpha}{2} + \frac{\sqrt{\alpha^2 - 4n\,\beta_1\,\beta_2}}{2} - g \\ \\ \frac{\alpha}{2} - \frac{\sqrt{\alpha^2 - 4n\,\beta_1\,\beta_2}}{2} - g \end{pmatrix}, \tag{2.53}$$

where $\alpha = \alpha_1 + (n-1)\alpha_2$. To ease readability, we will redefine A and G as follows:

$$A = \alpha_1 - g - \lambda \tag{2.54}$$
$$G = -g - \lambda \tag{2.55}$$

Proof. (i) For n=2: To extract the eigenvalues of the Jacobian, we set $det(\mathbf{J_{2s}} - \lambda\mathbf{I}) = 0$ and transform from Eq. 2.56 to Eq. 2.59 as follows:

$$det(\mathbf{J_{2s}} - \lambda\mathbf{I}) = 0 \tag{2.56}$$
$$\left| \begin{pmatrix} \alpha_1 - g & \alpha_2 & -\beta_1 \\ \alpha_2 & \alpha_1 - g & -\beta_1 \\ \beta_2 & \beta_2 & -g \end{pmatrix} - \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{pmatrix} \right| = 0 \tag{2.57}$$
$$\begin{vmatrix} \alpha_1 - g - \lambda & \alpha_2 & -\beta_1 \\ \alpha_2 & \alpha_1 - g - \lambda & -\beta_1 \\ \beta_2 & \beta_2 & -g - \lambda \end{vmatrix} = 0 \tag{2.58}$$
$$\begin{vmatrix} A & \alpha_2 & -\beta_1 \\ \alpha_2 & A & -\beta_1 \\ \beta_2 & \beta_2 & G \end{vmatrix} = 0 \tag{2.59}$$

Applying cofactor expansion to the first line of Eq. 2.59, we get Eq. 2.60 and can transform it to Eq. 2.62:

$$A \begin{vmatrix} A & -\beta_1 \\ \beta_2 & G \end{vmatrix} - \alpha_2 \begin{vmatrix} \alpha_2 & -\beta_1 \\ \beta_2 & G \end{vmatrix} - \beta_1 \begin{vmatrix} \alpha_2 & A \\ \beta_2 & \beta_2 \end{vmatrix} = 0 \tag{2.60}$$
$$A(AG + \beta_1\beta_2) - \alpha_2(\alpha_2 G + \beta_1\beta_2) - \beta_1\beta_2(\alpha_2 - A) = 0 \tag{2.61}$$
$$(A - \alpha_2)((A + \alpha_2)G + 2\beta_1\beta_2) = 0 \tag{2.62}$$

Substituting $\alpha = \alpha_1 + \alpha_2$, we get Eq. 2.63 and can subsequently transform it to Eq. 2.66:

$$(A - \alpha_2)((\alpha_1 - g - \lambda + \alpha_2)G + 2\beta_1\beta_2) = 0 \quad (2.63)$$

$$(A - \alpha_2)\{(\alpha - g - \lambda)(-g - \lambda) + 2\beta_1\beta_2\} = 0 \quad (2.64)$$

$$(A - \alpha_2)(\lambda^2 + (2g - \alpha)\lambda + g^2 - \alpha g - 2\beta_1\beta_2) = 0 \quad (2.65)$$

$$(A - \alpha_2)(\lambda - \frac{\alpha + \sqrt{\alpha^2 - 8\beta_1\beta_2}}{2} + g)(\lambda - \frac{\alpha - \sqrt{\alpha^2 - 8\beta_1\beta_2}}{2} + g) = 0 \quad (2.66)$$

From the factorized form of Eq. 2.66, we can directly read out the eigenvalues:

$$\lambda = \begin{pmatrix} \alpha_1 - \alpha_2 - g \\ \frac{\alpha}{2} + \frac{\sqrt{\alpha_1^2 - 8\,\beta_1\,\beta_2}}{2} - g \\ \frac{\alpha}{2} - \frac{\sqrt{\alpha^2 - 8\,\beta_1\,\beta_2}}{2} - g \end{pmatrix}, \quad (2.67)$$

where $\alpha = \alpha_1 + \alpha_2$. We can therefore conclude that Eq. 2.53 holds true for n=2.

(ii) Let us assume the induction hypothesis is true for $n = k$. Now, let n be $k+1$. Wishing to extract the eigenvalues, we can, again, set $det(\mathbf{J_{(k+1)s}} - \lambda\mathbf{I}) = 0$ and subsequently transform from Eq. 2.68 to Eq. 2.70:

$$det(\mathbf{J_{(k+1)s}} - \lambda\mathbf{I}) = 0 \quad (2.68)$$

$$\begin{vmatrix} \alpha_1 - g - \lambda & \alpha_2 & \cdots & \alpha_2 & -\beta_2 \\ \alpha_2 & \alpha_1 - g - \lambda & \cdots & \alpha_2 & -\beta_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_2 & \alpha_2 & \cdots & \alpha_1 - g - \lambda & -\beta_2 \\ \beta_1 & \beta_1 & \cdots & \beta_1 & -g - \lambda \end{vmatrix} = 0 \quad (2.69)$$

$$\begin{vmatrix} A & \alpha_2 & \cdots & \alpha_2 & -\beta_2 \\ \alpha_2 & A & \cdots & \alpha_2 & -\beta_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_2 & \alpha_2 & \cdots & A & -\beta_2 \\ \beta_1 & \beta_1 & \cdots & \beta_1 & G \end{vmatrix} = 0 \quad (2.70)$$

Carrying out cofactor expansion on the $(k+1)^{th}$, i.e., the second-to-last line of Eq. 2.70, Eq. 2.70 can be rewritten to Eq. 2.71:

$$
\pm\,\alpha_2
\begin{vmatrix}
\alpha_2 & \alpha_2 & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\
A & \alpha_2 & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\
\alpha_2 & A & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
\alpha_2 & \alpha_2 & \cdots & A & \alpha_2 & -\beta_2 \\
\beta_1 & \beta_1 & \cdots & \beta_1 & \beta_1 & G
\end{vmatrix}
\mp\,\alpha_2
\begin{vmatrix}
A & \alpha_2 & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\
\alpha_2 & \alpha_2 & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\
\alpha_2 & A & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
\alpha_2 & \alpha_2 & \cdots & A & \alpha_2 & -\beta_2 \\
\beta_1 & \beta_1 & \cdots & \beta_1 & \beta_1 & G
\end{vmatrix}
$$

$$
+ \cdots\cdots\cdots - \alpha_2 \underbrace{\begin{vmatrix}
A & \alpha_2 & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\
\alpha_2 & A & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
\alpha_2 & \alpha_2 & \cdots & A & \alpha_2 & -\beta_2 \\
\alpha_2 & \alpha_2 & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\
\beta_1 & \beta_1 & \cdots & \beta_1 & \beta_1 & G
\end{vmatrix}}_{*}
$$

$$
+ A
\begin{vmatrix}
A & \alpha_2 & \cdots & \alpha_2 & -\beta_2 \\
\alpha_2 & A & \cdots & \alpha_2 & -\beta_2 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
\alpha_2 & \alpha_2 & \cdots & A & -\beta_2 \\
\beta_1 & \beta_1 & \cdots & \beta_1 & G
\end{vmatrix}
- (-\beta_1)
\begin{vmatrix}
A & \alpha_2 & \cdots & \alpha_2 & \alpha_2 \\
\alpha_2 & A & \cdots & \alpha_2 & \alpha_2 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
\alpha_2 & \alpha_2 & \cdots & A & \alpha_2 \\
\beta_1 & \beta_1 & \cdots & \beta_1 & \beta_1
\end{vmatrix}
= 0 \quad (2.71)
$$

The signs of these expressions depend on the parity of (k+1). The sign for the cofactor of the $(k+1)^{th}$ element (the second-to-last one with coefficient A) is always positive as we do a cofactor expansion of the $(k+1)^{th}$ line and the sign is given by $(-1)^{(k+1)+(k+1)} = (-1)^{2(k+1)} = 1$. The adjacent elements therefore have negative signs.

When inspecting the cofactors with α_2 coefficients, we note that the overall set of rows is identical and that the i^{th} row of the i^{th} cofactor consists exclusively of α_2 entries. We can transform the determinant of the i^{th} to that of the k^{th} cofactor by iteratively swapping the row containing only α_2's with the one below until having reached the k^{th} row. With each swapping operation, the sign of the determinant changes. In the end, the determinant of the i^{th} cofactor will be identical to that of the k^{th} cofactor (the * part of 2.70) as the number of swaps is given by $k - i$, of which the parity matches with the sign difference with the k^{th} cofactor. Therefore Eq. 2.71 can be simplified to Eq. 2.72:

$$
\begin{aligned}
-k\alpha_2 \underbrace{\begin{vmatrix} A & \alpha_2 & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\ \alpha_2 & A & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \alpha_2 & \alpha_2 & \cdots & A & \alpha_2 & -\beta_2 \\ \alpha_2 & \alpha_2 & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\ \beta_1 & \beta_1 & \cdots & \beta_1 & \beta_1 & G \end{vmatrix}}_{*} \\
+A \underbrace{\begin{vmatrix} A & \alpha_2 & \cdots & \alpha_2 & -\beta_2 \\ \alpha_2 & A & \cdots & \alpha_2 & -\beta_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_2 & \alpha_2 & \cdots & A & -\beta_2 \\ \beta_1 & \beta_1 & \cdots & \beta_1 & G \end{vmatrix}}_{**} + \beta_1 \underbrace{\begin{vmatrix} A & \alpha_2 & \cdots & \alpha_2 & \alpha_2 \\ \alpha_2 & A & \cdots & \alpha_2 & \alpha_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_2 & \alpha_2 & \cdots & A & \alpha_2 \\ \beta_1 & \beta_1 & \cdots & \beta_1 & \beta_1 \end{vmatrix}}_{***} = 0 \quad (2.72)
\end{aligned}
$$

For the sake of readability and in order to ease comprehension, we will solve the three remaining determinants a parts of Eq. 2.72 successively.

Part 1 (*): Here, we will start by solving the first determinant of 2.72. By subtracting the k^{th} row, i.e., the second-to-last one, from the first row and applying cofactor expansion recursively, we derive the following:

$$
\begin{aligned}
\begin{vmatrix} A & \alpha_2 & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\ \alpha_2 & A & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \alpha_2 & \alpha_2 & \cdots & A & \alpha_2 & -\beta_2 \\ \alpha_2 & \alpha_2 & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\ \beta_1 & \beta_1 & \cdots & \beta_1 & \beta_1 & G \end{vmatrix} &= \begin{vmatrix} A-\alpha_2 & 0 & \cdots & 0 & 0 & 0 \\ \alpha_2 & A & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \alpha_2 & \alpha_2 & \cdots & A & \alpha_2 & -\beta_2 \\ \alpha_2 & \alpha_2 & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\ \beta_1 & \beta_1 & \cdots & \beta_1 & \beta_1 & G \end{vmatrix} \\
&= (A-\alpha_2) \begin{vmatrix} A & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ \alpha_2 & \cdots & A & \alpha_2 & -\beta_2 \\ \alpha_2 & \cdots & \alpha_2 & \alpha_2 & -\beta_2 \\ \beta_1 & \cdots & \beta_1 & \beta_1 & G \end{vmatrix} \\
&= \cdots \\
&= (A-\alpha_2)^{k-1} \begin{vmatrix} \alpha_2 & -\beta_2 \\ \beta_1 & G \end{vmatrix} \\
&= (A-\alpha_2)^{k-1}(\alpha_2 G + \beta_1\beta_2) \quad (2.73)
\end{aligned}
$$

Part 2 (**): The second determinant of 2.72 is the same as $det(\mathbf{J_{ks}} - \lambda\mathbf{I})$. Given the induction hypothesis, let us define:

$$\alpha_k = \alpha_1 + (k-1)\alpha_2 \tag{2.74}$$

It follows that:

$$det(\mathbf{J_{ks}} - \lambda\mathbf{I}) = (\alpha_1 - \alpha_2 - g - \lambda)^{k-1} * \left(\frac{\alpha_k + \sqrt{\alpha_k^2 - 4k\beta_1\beta_2}}{2} - g - \lambda\right)\left(\frac{\alpha_k - \sqrt{\alpha_k^2 - 4k\beta_1\beta_2}}{2} - g - \lambda\right) \tag{2.75}$$

$$= (A - \alpha_2)^{k-1} * \left(\frac{\alpha_k + \sqrt{\alpha_k^2 - 4k\beta_1\beta_2}}{2} + G\right)\left(\frac{\alpha_k - \sqrt{\alpha_k^2 - 4k\beta_1\beta_2}}{2} + G\right) \tag{2.76}$$

$$= (A - \alpha_2)^{k-1}\left(\left(\frac{\alpha_k}{2} + G\right)^2 - \left(\frac{\sqrt{\alpha_k^2 - 4k\beta_1\beta_2}}{2}\right)^2\right) \tag{2.77}$$

$$= (A - \alpha_2)^{k-1}\left(G^2 + \alpha_k G + k\beta_1\beta_2\right) \tag{2.78}$$

Part 3 (***): The third determinant of 2.72 can be calculated similarly to the first determinant. By substracting the column $(k+1)$ from the first one and applying cofactor expansion to the first column recursively, we get:

$$\begin{vmatrix} A & \alpha_2 & \cdots & \alpha_2 & \alpha_2 \\ \alpha_2 & A & \cdots & \alpha_2 & \alpha_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_2 & \alpha_2 & \cdots & A & \alpha_2 \\ \beta_1 & \beta_1 & \cdots & \beta_1 & \beta_1 \end{vmatrix} = \begin{vmatrix} A - \alpha_2 & \alpha_2 & \cdots & \alpha_2 & \alpha_2 \\ 0 & A & \cdots & \alpha_2 & \alpha_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \alpha_2 & \cdots & A & \alpha_2 \\ 0 & \beta_1 & \cdots & \beta_1 & \beta_1 \end{vmatrix} \tag{2.79}$$

$$= (A - \alpha_2)\begin{vmatrix} A & \cdots & \alpha_2 & \alpha_2 \\ \vdots & \ddots & \vdots & \vdots \\ \alpha_2 & \cdots & A & \alpha_2 \\ \beta_1 & \cdots & \beta_1 & \beta_1 \end{vmatrix} \tag{2.80}$$

$$= \cdots$$

$$= (A - \alpha_2)^{k-1}\begin{vmatrix} A & \alpha_2 \\ \beta_1 & \beta_1 \end{vmatrix} \tag{2.81}$$

$$= (A - \alpha_2)^k\beta_1 \tag{2.82}$$

Parts 1 - 3: To sum up the results obtained through transforming the three determinants from above, Eq. 2.72 is transformed as follows, from Eq. 2.83 to Eq. 2.86:

$$\begin{aligned} &- k\alpha_2(A-\alpha_2)^{k-1}(\alpha_2 G+\beta_1\beta_2) \\ &\quad + A(A-\alpha_2)^{k-1}\left(G^2+\alpha_k G+k\beta_1\beta_2\right)-\beta_2\beta_1(A-\alpha_2)^k=0 \end{aligned} \tag{2.83}$$

$$\begin{aligned} &(A-\alpha_2)^{k-1} \\ &\quad \left(A\left(G^2+\alpha_k G+k\beta_1\beta_2\right)-k\alpha_2(\alpha_2 G+\beta_1\beta_2)+\beta_1\beta_2(A-\alpha_2)\right)=0 \end{aligned} \tag{2.84}$$

$$A-\alpha_2)^{k-1}\left(G(AG+A\alpha_k-k\alpha_2^2)+(A-\alpha_2)(k+1)\beta_1\beta_2\right)=0 \tag{2.85}$$

$$\begin{aligned} &(A-\alpha_2)^{k-1} \\ &\quad \left(G\underbrace{(AG+A(\alpha_1+(k-1)\alpha_2)-k\alpha_2^2)}_{*}+(A-\alpha_2)(k+1)\beta_1\beta_2\right)=0 \end{aligned} \tag{2.86}$$

This expression can be further simplified. By definition of $A=\alpha_1+G$, the part of Eq. 2.86 denoted by * can be transformed to Eq. 2.87 and rewritten to Eq. 2.91:

$$\begin{aligned} (AG &+ A(\alpha_1+(k-1)\alpha_2)-k\alpha_2^2) \\ &= ((\alpha_1+G)G+(\alpha_1+G)(\alpha_1+(k-1)\alpha_2)-k\alpha_2^2) && (2.87)\\ &= (G^2+(2\alpha_1+(k-1)\alpha_2)G+\alpha_1^2+(k-1)\alpha_1\alpha_2-k\alpha_2^2) && (2.88)\\ &= (G^2+(2\alpha_1+(k-1)\alpha_2)G+(\alpha_1+k\alpha_2)(\alpha_1-\alpha_2)) && (2.89)\\ &= (G+\alpha_1+k\alpha_2)(G+\alpha_1-\alpha_2) && (2.90)\\ &= (G+\alpha_1+k\alpha_2)(A-\alpha_2) && (2.91) \end{aligned}$$

Therefore, all of Eq. 2.86 is equivalent to Eq. 2.92 which can be transformed to Eq. 2.95:

$$\begin{aligned} (A-\alpha_2)^{k-1}\left(G(G+\alpha_1+k\alpha_2)(A-\alpha_2)+(A-\alpha_2)(k+1)\beta_1\beta_2\right)&=0 && (2.92)\\ (A-\alpha_2)^{k}\left(G^2+(\alpha_1+k\alpha_2)G+(k+1)\beta_1\beta_2\right)&=0 && (2.93)\\ (A-\alpha_2)^{k}\left(G^2+(\alpha_1+((k+1)-1)\alpha_2)G+(k+1)\beta_1\beta_2\right)&=0 && (2.94)\\ (A-\alpha_2)^{k}\left(G^2+\alpha_{k+1}G+(k+1)\beta_1\beta_2\right)&=0 && (2.95) \end{aligned}$$

From Eq. 2.95, the eigenvalues for $\mathbf{J_{(k+1)s}}$ can be obtained:

$$\lambda = \begin{pmatrix} \left.\begin{matrix} \alpha_1 - \alpha_2 - g \\ \alpha_1 - \alpha_2 - g \\ \vdots \\ \alpha_1 - \alpha_2 - g \end{matrix}\right\} (k+1) - 1 \\ \\ \frac{\alpha_{k+1}}{2} + \frac{\sqrt{\alpha_{k+1}^2 - 4(k+1)\,\beta_1\,\beta_2}}{2} - g \\ \\ \frac{\alpha_{k+1}}{2} - \frac{\sqrt{\alpha_{k+1}^2 - 4(k+1)\,\beta_1\,\beta_2}}{2} - g, \end{pmatrix} \tag{2.96}$$

where $\alpha_{k+1} = \alpha_1 + ((k+1)-1)\alpha_2$.

From (i) and (ii), by the principle of induction, Eq. 2.53 is true for all integers $n \geq 2$.

□

We have shown that the eigenvalues of a Jacobian for a network with n excitatory and a single inhibitory unit, connected as shown in Fig. 2.7 B', are provided by Eq. 2.53. By conditioning the real part of those eigenvalues to be negative, the conditions for soft WTA of (n+1) units can be obtained:

Case I: $\alpha^2 - 4n\beta_1\beta_2 < 0$:

$$\alpha_1 - \alpha_2 < g \tag{2.97}$$
$$\alpha < 2g \tag{2.98}$$

Case II: $\alpha^2 - 4n\beta_1\beta_2 \geq 0$:

$$\alpha_1 - \alpha_2 < g \tag{2.99}$$
$$\alpha < 2g \tag{2.100}$$
$$\beta_1\beta_2 > \frac{1}{n}(\alpha - g)g, \tag{2.101}$$

where $\alpha = \alpha_1 + (n-1)\alpha_2$.

To conclude, we were able to derive conditions for stable soft WTA behavior in networks of n excitatory neurons and a single inhibitory neuron, assuming the connectivity depicted in Fig. 2.7. Providing conditions for differently connected network will be subject of future investigations.

2.4 Hysteresis and self-sustained behavior

2.4.1 Introduction

So far, we focussed on deriving conditions for stability in soft and hard winner-take-all configurations. Here, we wish to derive conditions to account for two further phenomena: hysteresis and self-sustained behavior. Hysteresis describes the dependence of a system not only on its most recent input but also on its history. More technically, a system that exhibits hysteresis is characterized by a bistable region where, depending on its history, one of (at least) two different states is taken.

To illustrate this idea, Fig. 2.8 depicts an exemplary phase portrait of a system with one excitatory (u_1) and one inhibitory (u_2) unit. The x-axis thereby represents the activity of the excitatory unit while the y-axis represents its scaled time derivative. Points that fall on the dashed black line are fixed points. The dynamics of the system are depicted for different external input strengths.

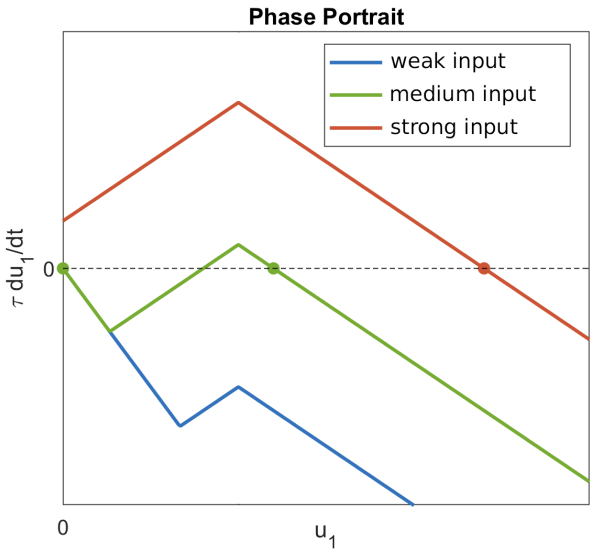


Figure 2.8: Exemplary phase portrait of 2-unit network. Line colors represent external input strengths, ranging from weak (blue) to strong (red).

The key property of hysteresis is that the system, at a stimulus level similar to the green line, will converge to either the left or the right fixed point, depending on the history of the system: When starting without external input, the dynamics of the system are given by the blue line in Fig. 2.8, which only has a single fixed point at $u_1 = 0$. If the input strength is increased to the level of the green line, the system will remain at the fixed point at $u_1 = 0$, even though a second, higher, fixed point is present. When increasing the input strength until reaching the level of the red line, its single fixed point ($u_1 > 0$) will be taken. Importantly, when

decreasing the stimulus strength again to that of the green line, the higher fixed point will be taken ($u_1 > 0$). This way, the converging state of the system with medium-strong input depends on the history of the input: coming from weaker input, the lower fixed point will be taken; coming from stronger input, the higher fixed point will be taken.

After having introduced and discussed hysteresis, a second type of configuration we wish to investigate in this context is that of self-sustained behavior. Self-sustained behavior describes the phenomenon of a system that, once sufficiently activated by an external force, continues to be active even in the absence of that force. More visually, self-sustained behavior holds if the line of the phase portrait that is corresponding to no external input has a positive second turning point. In this case, the activity does not fall back to zero once the external input is absent.

While self-sustained behavior could, given its history dependency, technically be considered a special form of hysteresis, we differentiate between hysteresis and self-sustained behavior by the excitability in the absence of input following sufficiently strong stimulation. We therefore regard hysteresis and self-sustained behavior as mutually exclusive. Technically, we differentiate between them by assessing the sign of the second turning point when the input level is zero.

Several conditions need to hold in case of both self-sustained behavior and hysteresis that is not self-sustained: When inspecting Fig. 2.8, we see that all depicted functions are made up of three sub-functions. Crucially, the gradient of the first sub-function has to be negative, that of the second one positive, and that of the third one negative. The two turning points further need to hold positive x-values, and a certain amount of stimulus strength is required. Furthermore, some conditions derived in the context of WTA stability need to hold in order to prevent unstable behavior.

2.4.2 Phase portrait and derivation of conditions

After having introduced hysteresis and self-sustained behavior conceptually, we will now derive the conditions for corresponding parameter values. Here, we will start with the most simple meaningful network structure possible: a network consisting of one excitatory (u_1) and one inhibitory unit (u_2). The units' activity can be described as follows, adapted from Rutishauser et al. (2011):

$$\tau \frac{du_1}{dt} = -gu_1 + F(s + \alpha u_1 - \beta_2 u_2 - T_1) \tag{2.102}$$

$$\tau \frac{du_2}{dt} = -gu_2 + F(\beta_1 u_1 - T_2), \tag{2.103}$$

where $F(x) = max(x, 0)$, $s \geq 0$ is the input to the excitatory unit, and g is the leak conductance. α_1, β_2, and β_1 are non-negative weights.

We begin by deriving the conditions under which the inhibitory neuron (u_2) reaches its equilibrium. Setting $\frac{du_2}{dt} = 0$, we can reformulate Eq. 2.103:

$$u_2 = \frac{F(\beta_1 u_1 - T_2)}{g} \tag{2.104}$$

Next, we can carry out a case differentiation:

Case I: When $\beta_1 u_1 - T_2 < 0 \iff u_1 < \frac{T_2}{\beta_1}$:

$$u_2 = 0 \tag{2.105}$$

Plugging Eq. 2.105 into Eq. 2.102 yields:

$$\tau \frac{du_1}{dt} = -g u_1 + F(s + \alpha_1 u_1 - T_1) \tag{2.106}$$

Note that Eq. 2.106 with $F(x) = max(x, 0)$ is 0 if $u_1 < \frac{T_1 - s}{\alpha_1}$. We can subsequently summarize Eq. 2.106 as follows:

$$\tau \frac{du_1}{dt} = \begin{cases} -g u_1 & u_1 < \frac{T_1 - s}{\alpha_1} \\ (\alpha_1 - g) u_1 + s - T_1 & \frac{T_1 - s}{\alpha_1} \leq u_1 \end{cases} \tag{2.107}$$

Case II: When $\beta_1 u_1 - T_2 \geq 0$:

Plugging Eq. 2.104 into Eq. 2.102 yields:

$$\tau \frac{du_1}{dt} = (\alpha_1 - g) u_1 + s - T_1 - \beta_2 \frac{\beta_1 u_1 - t2}{g} \tag{2.108}$$

Case I+II: To sum up both cases:

$$\tau \frac{du_1}{dt} = \begin{cases} -g u_1 & u_1 < \frac{T_1 - s}{\alpha_1} \\ (\alpha_1 - g) u_1 + s - T_1 & \frac{T_1 - s}{\alpha_1} \leq u_1 < \frac{T_2}{\beta_1} \\ (\alpha_1 - g) u_1 + s - T_1 - \beta_2 \frac{\beta_1 u_1 - t2}{g} & \frac{T_2}{\beta_1} \leq u_1 \end{cases} \tag{2.109}$$

Hysteresis and self-sustained behavior: We now solve these equations under the conditions that hold in both hysteresis and self-sustained behavior: (i) having two turning points in a positive region and having (ii) a negative gradient in the first, (iii) a positive gradient in the second, and (iv) a negative gradient in the third of the equations in Eq. 2.109.

In order to have two turning points in a positive region ($u_1 > 0$), it has to hold that:

$$T_1 > 0 \tag{2.110}$$

$$T_2 > 0 \tag{2.111}$$

The gradient of the first of the equations in Eq. 2.109 is trivially negative. The gradient of the second of the equations in Eq. 2.109 with respect to u_1 has to be positive. Therefore, it has to hold that:

$$g < \alpha_1 \tag{2.112}$$

Finally, the gradient of the third of the equations in Eq. 2.109 has to be negative. This holds true if:

$$\alpha_1 - g - \frac{\beta_1 \beta_2}{g} < 0 \tag{2.113}$$

$$\beta_1 \beta_2 > (\alpha_1 - g) g \tag{2.114}$$

These conditions are, in fact, identical to those derived for hard WTA stability. This makes sense as we are currently considering the interaction between a single excitatory and a single inhibitory unit.

There are a few additional constraints that have to be taken into account. From the convergence condition (the stability of hard WTA discussed in the last sections can be used in this context due to the two-unit interactions), an upper limit of α_1 is set to $2g$:

$$\alpha_1 < 2g \tag{2.115}$$

In Fig. 2.8, we further see that only a certain range of external stimulation would allow two stable fixed points. This range can be obtained by calculating the value of the second turning point being positive ($\tau \frac{du_1}{dt} > 0$) and the first turning point being negative ($\tau \frac{du_1}{dt} < 0$), which is as follows:

$$(\alpha_1 - g) \frac{T_2}{\beta_1} + s - T_1 > 0 \tag{2.116}$$

$$-g \frac{T_1 - s}{\alpha_1} < 0 \tag{2.117}$$

Therefore, the following has to hold:

$$T_1 - (\alpha_1 - g)\frac{T_2}{\beta_1} < s < T_1 \tag{2.118}$$

We have seen that in order to observe hysteresis or self-sustained behavior, the external input s has to change, covering the range given above. For instance, if we start with $s = 0$, s has to exceed T_1 in order for the system to take the upper fixed point.

In addition to all conditions mentioned above, additional conditions from contraction theory, derived in the previous sections, are applied in order to avoid, for instance, explosive behavior. At the end of this section, all conditions that need to hold for a system to show hysteresis or exhibit self-sustained behavior, are provided.

Hysteresis: Hysteresis, on the one hand, and self-sustained behavior, on the other hand, share several properties. To derive further conditions, we here need to differentiate between them. We will here begin with the derivation of conditions for hysteresis. A condition that needs to be fulfilled to have hysteresis not self-sustain is that the second turning point is negative when s = 0, such that the activity falls back to zero in the absence of any input. Plugging $u_1 = \frac{T_2}{\beta_1}$ into the third of the equations in Eq. 2.109 yields Eq. 2.119 and can be transformed to Eq. 2.120:

$$(\alpha_1 - g)\frac{T_2}{\beta_1} + 0 - T_1 < 0 \tag{2.119}$$

$$\alpha_1 < \frac{T_1}{T_2}\beta_1 + g \tag{2.120}$$

When $T_1 > 0$ (see condition i in Eq. 2.109), $T_2 > 0$ (see condition ii in Eq. 2.109), and β_2 and g are given, the parameter range for α_1 in which the network shows hysteresis can, using Equations 2.112 and 2.120, be expressed as follows:

$$g < \alpha_1 \leq \frac{T_1}{T_2}\beta_1 + g \tag{2.121}$$

Self-sustained behavior: Self-sustained behavior, on the other hand, can be derived by enforcing that the second turning point be positive when $s = 0$. In a manner analogous to the derivation of the hysteresis condition, we can derive Eq. 2.122. Taken together, we can summarize the conditions for hysteresis (Eq. 2.123-2.125) and self-sustained behavior (Eqs. 2.126-2.128).

$$\frac{T_1}{T_2}\beta_1 + g < \alpha_1 < 2g \tag{2.122}$$

Hysteresis range:

$$g < \alpha_1 \leq \frac{T_1}{T_2}\beta_1 + g \tag{2.123}$$

$$(\alpha_1 - g)g < \beta_1\beta_2 \tag{2.124}$$

$$T_1 - (\alpha_1 - g)\frac{T_2}{\beta_1} < s < T_1 \tag{2.125}$$

Self-sustained range:

$$\frac{T_1}{T_2}\beta_1 + g < \alpha_1 < 2g \tag{2.126}$$

$$(\alpha_1 - g)g < \beta_1\beta_2 \tag{2.127}$$

$$0 < s < T_1 \tag{2.128}$$

These analytical results we were able to validate through numerical simulations that showed that parameter sets fulfilling these conditions would, indeed, lead to the specified type of desired behavior (not shown here).

2.4.3 Phase plane and prediction of activity

An alternative way of describing the dynamics of a system is through the depiction of its phase plane (for a mathematical introduction, see, for instance, Chapter 4.3 of Gerstner et al. (2014)). The left panel of Fig. 2.9 shows the phase plane of one excitatory and one inhibitory neuron: the red line represents the nullcline of the differential equation governing the excitatory unit; the blue line the nullcline of the inhibitory unit. The crossings of these nullclines are the fixed points of the system. Under the parameter condition used for producing Fig. 2.9, three fixed points, out of which two are stable, can be found.

Through analysis of these stable fixed points, we can predict the activity of the system. This is illustrated in the right panel of Fig. 2.9: it shows the two units' activity as a function of the external input to the excitatory unit. In the parameter setting given, the hysteresis property becomes apparent: the u_1 forward line (resulting from gradually increasing the input strength) and the u_2 backward line (resulting from gradually decreasing the input strength) differ for a part of the input levels. Hence, depending on whether coming from lower or higher activity level, different fixed points are taken.

For the purpose of further illuminating the dynamics of this system, we built an interactive app allowing a user to enter the parameter values for a network into the GUI and explore its dynamics (see Appendix A).

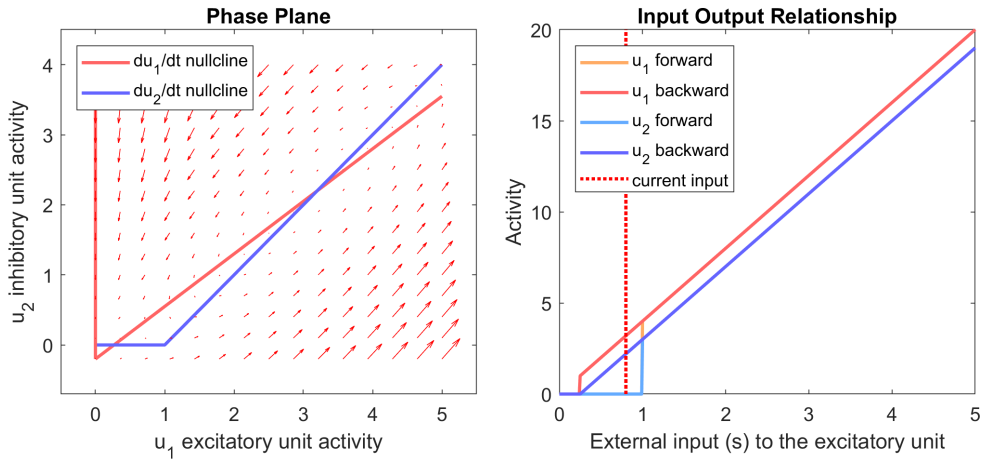


Figure 2.9: Depiction of phase plane and corresponding input-output relationship for 2-unit network with $\alpha_1 = 0.75$, $\beta_1 = \beta_2 = 1$, $g_1 = g_2 = 1$, $T_1 = T_2 = 1$, $s = 0.8$. Left panel: Nullclines of the differential equation governing the excitatory (red) and inhibitory (blue) unit are plotted. The intersection of nullclines represent fixed points of the system. Arrows represent the vector field of the direction and gradient of activity changes at each grid point. Right panel: A unit's activity as a function of external input to the excitatory unit. The dashed red line represents the current external input (always corresponding to the one shown in the left panel as phase plane). Figures are created using our interactive Matlab app (see Appendix A).

2.5 Overview of different behavior classes

Thus far, we extended the stability analysis of soft and hard WTA networks to arbitrarily large networks and derived novel conditions for hysteresis as well as self-sustained behavior. We can now put it all together.

Fig. 2.10 visualizes parameter regions associated with different winner-take-all behaviors in a simple network of two excitatory and a single inhibitory unit, as depicted in Fig.2.1 A. Panels (1)-(3) each depict the inputs to the excitatory units as well as the activity of each unit. The different points in the parameter space thereby represent examples for (1) soft WTA, (2) hard WTA, and (3) unstable behavior.

Fig. 2.11, for a network consisting of one excitatory and one inhibitory unit, visualizes parameter regions associated with different classes of behaviors: (1) regular non-sustained activity without hysteresis (2) hysteresis, (3) self-sustained behavior, and (4) unstable behavior. For each of the four types, similar to Fig. 2.9, phase planes as well as the relationship between external inputs to the excitatory neuron and the activity of both inhibitory and excitatory neuron are depicted.

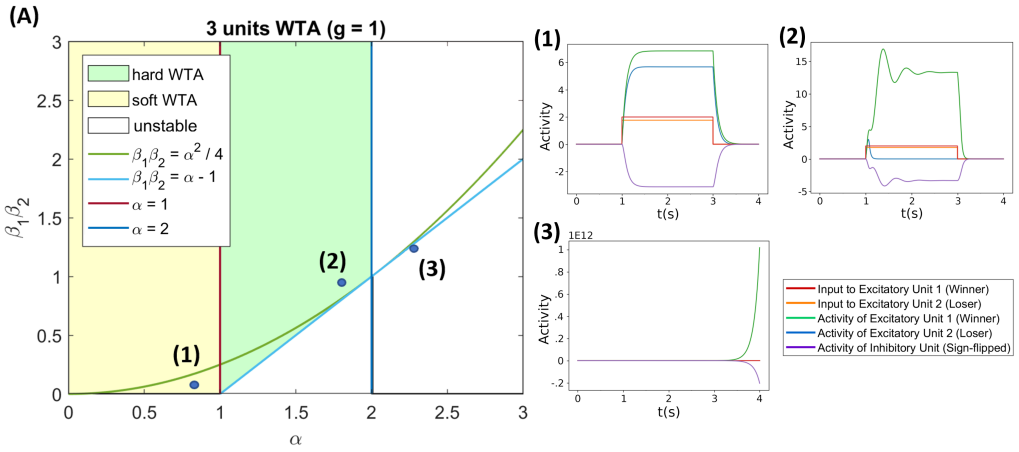


Figure 2.10: Panel (A) Visualization of different parameter regions associated with different WTA behaviors. The leak parameter g was set to 1. Panel (1)-(3) are some exemplar numerical simulations for the parameter set shown as dots in (A). The first excitatory unit represents the winner, which receives slightly larger input than the second loser unit. The activity of the inhibitory unit was sign-flipped for the sake of visualization, such that positive activity of the inhibitory unit is plotted below zero. Panel (1): A soft winner-take-all example with parameters $\alpha_1 = 0.8$, $\beta_1\beta_2 = 0.05$. Panel (2): A hard winner-take-all example with $\alpha_1 = 1.8$, $\beta_1\beta_2 = 0.95$. Panel (3): An unstable example with parameters $\alpha_1 = 2.25$, $\beta_1\beta_2 = 1.25$.

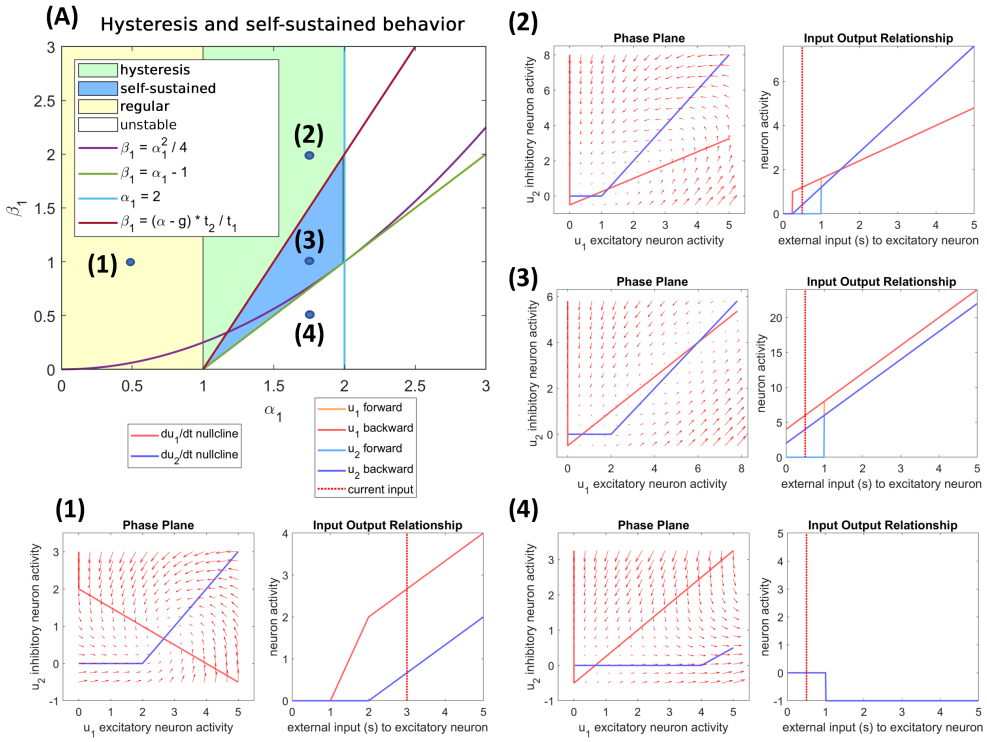


Figure 2.11: Panel (A) Visualization of parameter regions associated with different hysteresis and self-sustained behaviors. The fixed parameters were as follows: $\beta_2 = 1$, $g = 1$, $T_1 = 1$, $T_2 = 2$. Left subpanel of panels (1)-(4): Phase plane of the activity of an excitatory and an inhibitory neuron. The red line indicates the nullcline for the excitatory unit, the blue line the nullcline for the inhibitory unit. The intersections of those nullclines determine the fixed points. Right subpanel of panels (1)-(4): Relationship between external inputs to the excitatory neuron and the activity of the inhibitory and excitatory neuron. Explosion (no valid fixed point) is expressed as neuronal activity of -1 on the y-axis of the right panel.

Chapter 3

Spiking model dynamics

In the last chapter, we carried out a stability analysis for soft and hard WTA networks and derived conditions for hysteresis and self-sustained behavior. In this chapter, we aim to translate some of those dynamics to networks of spiking model neurons. We will herein start by an analysis of individual neurons and later extend our analysis to groups of neurons.

3.1 Dynamics of individual model neurons

3.1.1 Introducing the leaky-integrate-and-fire model

A popular choice among simplified spiking neuron models is the leaky-integrate-and-fire (LIF) neuron. This type of neuron accumulates input and generates a spike when exceeding a certain threshold (Gerstner and Kistler, 2002). The temporal evolution of the neuron's membrane potential v is given by:

$$\tau \frac{dv}{dt} = -(v - v_{rest}) + RI, \tag{3.1}$$

where τ represents the membrane time constant of the neuron, v the membrane potential, and R and I the neuron's resistance and input current, respectively. Upon exceeding a threshold (v_θ), the neuron generates a spike and its membrane potential is set to the resting potential v_{rest}:

$$v \leftarrow v_{rest} \; when \; v \geq v_\theta \tag{3.2}$$

We first wish to consider a scenario with constant input (I_0). By setting the resetting time $t_0 = 0$, and defining t_{ref} as the refractory period, the temporal evolution of v can be described as follows:

$$v(t) = v_{rest} + RI_0 \left(1 - \exp\left(-\frac{t - t_{ref}}{\tau}\right)\right), \tag{3.3}$$

where $t > t_{ref}$.

The time, t_θ, that indicates when v(t) reaches its threshold, v_θ, is calculated as follows:

$$v_\theta = v_{rest} + RI_0 \left(1 - \exp\left(-\frac{t_\theta - t_{ref}}{\tau}\right)\right) \tag{3.4}$$

$$t_\theta = t_{ref} - \tau \ln\left(1 - \frac{v_\theta - v_{rest}}{RI_0}\right) \tag{3.5}$$

The firing frequency, f, of an LIF neuron can therefore be expressed as:

$$f = \frac{1}{t_\theta} = \frac{1}{t_{ref} - \tau \ln\left(1 - \frac{v_\theta - v_{rest}}{RI_0}\right)} \tag{3.6}$$

3.1.2 Relationship between input and output frequency

3.1.2.1 Analytical derivation

When thinking about single neuron behavior, knowing the relationship between presynaptic and postsynaptic firing frequencies can serve as a substitute for the activation function (e.g. ReLU in the model by Rutishauser et al. (2011)).

When the neuron receives an input voltage – this is how the simulations are carried out – its membrane potential increases but afterwards decays exponentially due to the leak component. Let n_s be the number of spikes that are required for the postsynaptic neuron to spike, let t_{ref} be its refractory period, and w the intensity of the synaptic input. When assuming that the firing frequency of the presynaptic neuron, f_{pre} (Hz), is constant, the interval between spikes is also constant and equal to $\frac{1}{f}$ (s). Let a_n now be the voltage level present immediately before receiving the next spike. At the arrival of the next spike, the voltage becomes $a_n + w$ and decays until the reception of the next spike in $\frac{1}{f}$ (s). Therefore, by using a_n, a_{n+1}, can be expressed as Eq. 3.7. Fig. 3.1 illustrates how to determine the number of spikes that the postsynaptic neuron needs to receive from the presynaptic neuron in order to spike.

$$a_{n+1} = (a_n + w) \exp\left(-\frac{1}{f\tau}\right) \tag{3.7}$$

We will now carry out a series of transformations in order to derive an equation of the postsynaptic firing rate. When $p = \exp\left(-\frac{1}{f\tau}\right)$ and $q = \frac{wp}{1-p}$, we can reformulate Eq. 3.7 to Eq. 3.8 and transform it to Eq. 3.11 as follows:

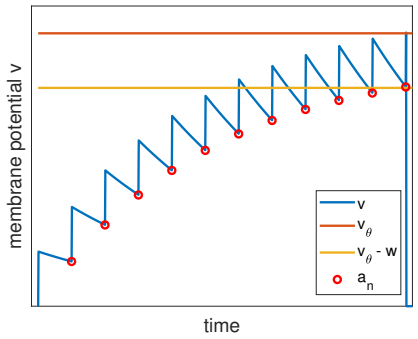


Figure 3.1: Illustration of how to determine the number of spikes the postsynaptic neuron needs to receive in order to generate a spike. v represents the neuron's membrane potential, a_n captures the membrane potential immediately before receiving and integrating the next spike, v_θ is the threshold above which the neuron spikes, and w is the synaptic input intensity.

$$
\begin{aligned}
a_{n+1} - q &= (a_n - q)p && (3.8)\\
a_n &= (a_1 - q)p^{n-1} + q && (3.9)\\
a_1 &= wp && (3.10)\\
a_n &= \frac{wp - wp^{n+1}}{1-p} && (3.11)
\end{aligned}
$$

When n_s is the number of input spikes that are required for the postsynaptic neuron to spike, the number of input spikes required to reach $\Delta v - w$, n', are equal to $n' = n_s - 1$. It follows that:

$$
a_{n'} = \frac{wp - wp^{n'+1}}{1-p} \geq \Delta v - w \tag{3.12}
$$

Since $p < 1$, it holds that:

$$
\begin{aligned}
wp - wp^{n'+1} &\geq (\Delta v - w)(1-p) && (3.13)\\
wp^{n'+1} &\leq wp - (\Delta v - w)(1-p) && (3.14)\\
\ln(w) - \frac{n'+1}{f_{pre}\tau} &\leq \ln(\Delta v p - \Delta v + w) = \ln(w - \Delta v(1-p)) && (3.15)\\
n'+1 &\geq f_{pre}\tau\left(\ln(w) - \ln\left(w - \Delta v(1-p)\right)\right) && (3.16)\\
&= -f_{pre}\tau \ln\left(1 - \frac{\Delta v}{w}\left(1 - \exp\left(-\frac{1}{f_{pre}\tau}\right)\right)\right) && (3.17)\\
n'+1 &= \left\lceil -f_{pre}\tau \ln\left(1 - \frac{\Delta v}{w}\left(1 - \exp\left(-\frac{1}{f_{pre}\tau}\right)\right)\right)\right\rceil && (3.18)
\end{aligned}
$$

Since $n' = n_s - 1$:

$$n_s = \left\lceil -f_{pre}\tau \ln\left(1 - \frac{\Delta v}{w}\left(1 - \exp\left(-\frac{1}{f_{pre}\tau}\right)\right)\right)\right\rceil \tag{3.19}$$

We can therefore describe the postsynaptic firing rate as follows:

$$f_{post} = \begin{cases} \frac{1}{\frac{n_s}{f_{pre}} + t_{ref}} & f_{pre} > f_\theta \\ 0 & f_{pre} \leq f_\theta \end{cases} \tag{3.20}$$

$$f_\theta = -\frac{1}{\tau \ln\left(1 - \frac{w}{\Delta v}\right)} \tag{3.21}$$

$$w \leq \Delta v \tag{3.22}$$

$$\Delta v = v_\theta - v_{rest} \tag{3.23}$$

3.1.2.2 Validation through simulation

To validate the analytical results, we carried out simulations and compared the relation between presynaptic and postsynaptic firing rates obtained through derivation and simulation. To this end, we simulated a single LIF neuron for 2 seconds (in simulation time) for each input firing frequency of interest. The input was thereby chosen to be regular, as assumed in the analytical case. The postsynaptic firing rate was calculated as the average firing frequency in the specified time interval. All simulations were conducted using Brian 2 (Goodman and Brette, 2008). Fig. 3.2 depicts the resulting FF-curves (the postsynaptic firing rate as a function of presynaptic firing rate) obtained through simulations and analytical derivation. A comparison reveals the strong correspondence between theory and simulation.

3.1.2.3 Linearity of F-F curves as a function of weight

We have seen in the F-F curves depicted in Fig. 3.2 that, in case of a low input frequency and a large weight, the effect of the ceiling function is strong and the relationship between presynaptic and postsynaptic firing non-smooth. It is important to note here that the parameter set used for the derivation and simulation is characterized by a strong weight parameter. When the weight is sufficiently small compared to Δv and the input frequency sufficiently large, instead, the effect becomes negligible, as is shown in Fig. 3.3. We further observed and validated those results in spiking simulations (results not shown here). Given the observed linearity, we can now simplify Eq. 3.20:

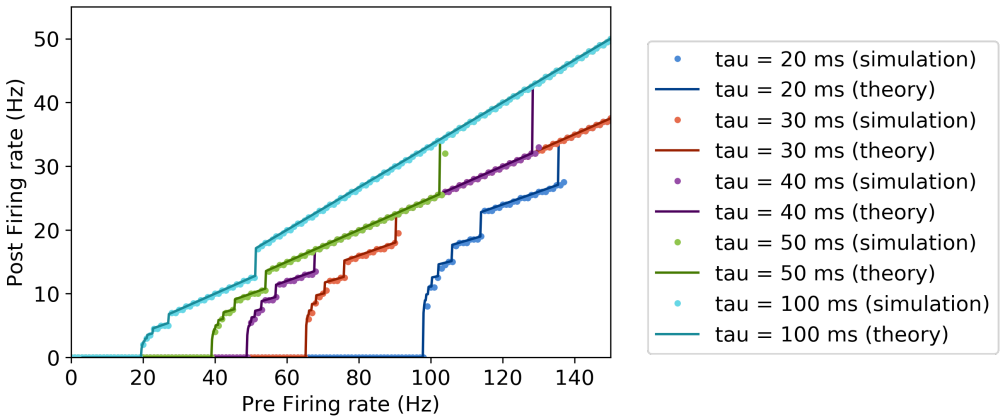


Figure 3.2: Comparison of analytical derivation and simulation results. F-F curves, depicting the postsynaptic firing rate as a function of the presynaptic firing rate, are given for both theory and simulation. Lines thereby represent analytical results; dots the simulation results. Different colors represent different time constants. Further parameters are given by $v_\theta = 1, v_{rest} = 0, w = 0.4$.

$$f_{post} = \begin{cases} \frac{1}{t_{ref} - \tau \ln\left(1 - \frac{\Delta v}{w}\left(1 - \exp\left(-\frac{1}{f_{pre}\tau}\right)\right)\right)} & f_{pre} > f_\theta \\ 0 & f_{pre} \leq f_\theta \end{cases} \tag{3.24}$$

$$f_\theta = -\frac{1}{\tau \ln\left(1 - \frac{w}{\Delta v}\right)} \tag{3.25}$$

$$w \leq \Delta v \tag{3.26}$$

$$\Delta v = v_\theta - v_{rest} \tag{3.27}$$

It is worth pointing out that there is a correspondence between the F-F curves (postsynaptic firing frequency as a function of presynaptic firing frequency) and I-F curves (postsynaptic firing frequency as a function of the input current). Specifically, by comparing Eq. 3.6 for the IF-curves and Eq. 3.24 for FF-curves, we see that they are equal under the following condition:

$$RI_0 = \frac{w}{1 - \exp\left(-\frac{1}{f_{pre}\tau}\right)} \tag{3.28}$$

Note that this is applicable when $RI_0 > \Delta v$, due to the valid range of the equations that are used in the comparison.

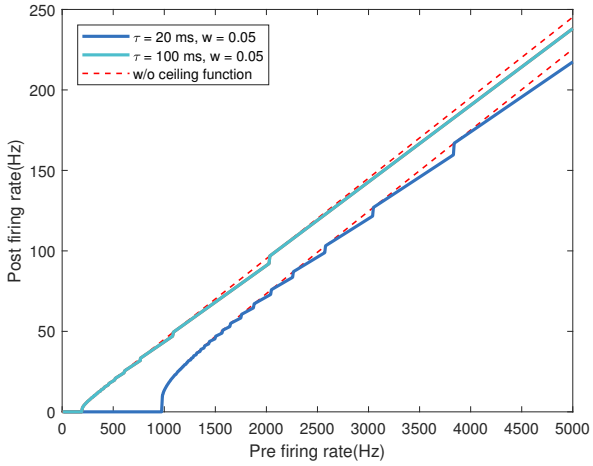


Figure 3.3: Analytically derived F-F curves for small weights and high input frequencies. The blue and light blue lines are calculated with a ceiling functions; the orange dotted lines are calculated without. For the configuration shown, the differences are marginal.

3.1.2.4 Linearization by series expansion

When $t_{ref} = 0$, a Laurent series expansion of Eq. 3.24 at $f_{pre} = \infty$ yields:

$$f_{post} = -\frac{1}{\tau \ln\left(1 - \frac{\Delta v}{w}\left(1 - \exp\left(-\frac{1}{f\tau}\right)\right)\right)} \tag{3.29}$$

$$= \frac{w}{\Delta v} f + \frac{w - \Delta v}{2\tau \Delta v} + \frac{w^2 - (\Delta v)^2}{12 f \tau^2 \Delta v w} + \frac{\Delta v (w - \Delta v)}{24 f^2 \tau^3 w^2} + \cdots \tag{3.30}$$

Here, f is short for f_{pre}. Note that this expansion is at $f_{pre} = \infty$ and f_{pre} should generally be large, so the third and later component approaches zero and can therefore be neglected. What remains is the following:

$$f_{post} \simeq \frac{w}{\Delta v} f_{pre} + \frac{w - \Delta v}{2\tau \Delta v} \tag{3.31}$$

Hence, when f_{pre} is large, the F-F function is linear with a gradient of $\frac{w}{v}$. When we look at the gradient of this function itself, we find that convergence is very fast (see Fig. 3.4). The derivative of the final firing rate input-output

function is provided by:

$$\frac{df_{post}}{df_{pre}} = \frac{\Delta v \exp\left(-\frac{1}{f\tau}\right)}{f^2\tau^2 w\left(1-\frac{\Delta v}{w}\left(1-\exp\left(-\frac{1}{f\tau}\right)\right)\right)\ln^2\left(1-\frac{\Delta v}{w}\left(1-\exp\left(-\frac{1}{f\tau}\right)\right)\right)} \tag{3.32}$$

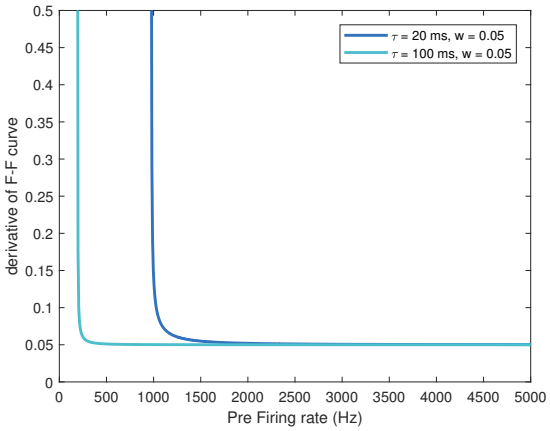


Figure 3.4: The derivative of the F-F function (Eq. 3.32) for $\tau = 20$ ms, 100 ms is plotted. The gradient of the F-F function converges to $\frac{w}{\Delta}v$ rapidly.

3.2 From single neurons to groups

Thus far in this chapter, we introduced the leaky-integrate-and-fire neuron model and described the relationship between the input and output firing rate for a single LIF neuron under regular spiking input. Wishing to study winner-take-all dynamics and the role of self-excitation, which cannot plausibly be modeled with a single neuron due to the implausibility of self-excitation in a single neuron, we will now turn to the analysis of a network of several LIF neurons.

In order to make use of the careful assessment presented in the previous chapter, which allows linking parameters of a rate-based model to different types of behaviors, we here seek to relate those findings to the spiking model. This bridge can be built in the general context of the mean-field approach which allows reducing the activity patterns in a network of neurons to average firing rates. While Wilson and Cowan (1972) and Amari (1977) were the first to employ this approach, it has been applied in numerous contexts and for many models. Among those models is the (leaky) integrate-and-fire neuron model (Brunel, 2000, Gerstner, 2000).

Before turning to the analysis of networks of neurons, there are a few methodological considerations that we wish to make beforehand.

3.2.1 Using Poisson spike trains as input

Neurons in our network model will not only receive input from external neurons but also from recurrently connected neighboring neurons. This makes the temporal arrival of input spikes less regular and the formalization as a random process plausible. In keeping with the observation that spike trains can often be described as a Poisson process (Heeger, 2000), we will assume a Poisson input for our simulations.

Fig. 3.5 depicts a comparison of the theoretically derived F-F curve, which is based on regular input, and the result of simulations which make use of non-regular Poisson input. We see that the two curves differ only for a presynaptic firing frequency that is just below the threshold. This is intuitively plausible: as the membrane potential is fluctuating in the subthreshold regime, some fluctuation in the spike time interval can make the neuron fire. This does not apply to the theoretically derived curve.

Instead of deriving a new F-F function for Poisson input, we will, given the strong correspondence, utilize the function that was derived in the context of regular external input.

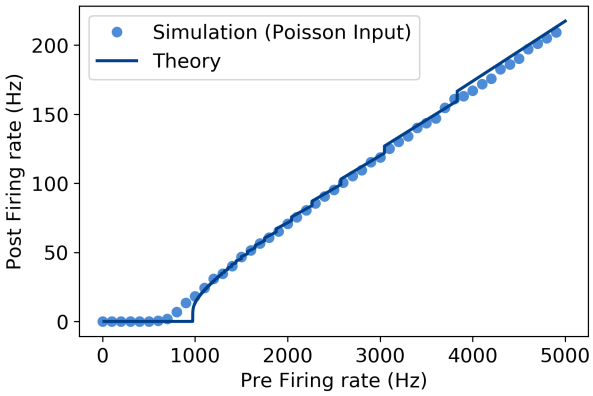


Figure 3.5: Comparison of theoretical F-F curve and simulation results with Poisson input. The following parameters were used: weight $= 0.05$, $\tau = 20$ ms. For each data point in the simulation, the simulation was carried out for 5 seconds (in simulation time). Simulations were carried out in Brian 2 (Goodman and Brette, 2008).

3.2.2 Combining different weights and frequencies

So far, we derived an F-F function in the context of single neurons with single weights and frequencies. In a group of neurons, we are confronted with differences between the synaptic weights for external input, excitatory input, and inhibitory

input (where even the sign is inverted), as well as differences in their corresponding firing frequencies. Therefore, we wish to calculate an appropriate average of these weights and firing frequencies.

Suppose one neuron is receiving inputs from N different sources (w_1, f_1), (w_2, f_2), $\cdots$, (w_N, f_N), where w_i and f_i represent the weight and frequency of the i^{th} input neuron. One way to calculate the combined weight w' and input frequency f' would be a firing-rate weighted averaged w and a summed up f, formalized as follows:

$$w' = \sum^{N} \frac{f_i}{f'} w_i \tag{3.33}$$

$$f' = \sum^{N} f_i \tag{3.34}$$

A further consideration is that if one neuron is connected to a group of N neurons with a connection probability of p, each of which is firing at f (Hz) on average, then the input frequency to the neuron is:

$$f' = Npf. \tag{3.35}$$

To validate Eqs. 3.33 and 3.34, we engaged in a simulational study. We took one neuron and two Poisson sources providing spiking inputs to the neuron. For the Poisson sources, we thereby randomly picked the firing frequencies as well as the weights for the connection from the input to the neuron (ranging from positive to negative, representing excitatory to inhibitory connections). We ran a simulation and measured the average firing frequency of the output neuron. Next, we calculated the w' and f' for this neuron, according to Eqs. 3.33 and 3.34. Then, we engaged in a second simulation where we utilized one neuron and a single Poisson neuron as input to that neuron. The input weight and frequency was set to w' and f'. If our formalism for combining multiple weights and frequencies is working properly, those two simulations should yield the same output frequencies. Fig. 3.6 shows a scatter plot of 1000 trials of such simulation sets. The results reveal that the relation is strongly linear. We are therefore going to use this summary description for our groups.

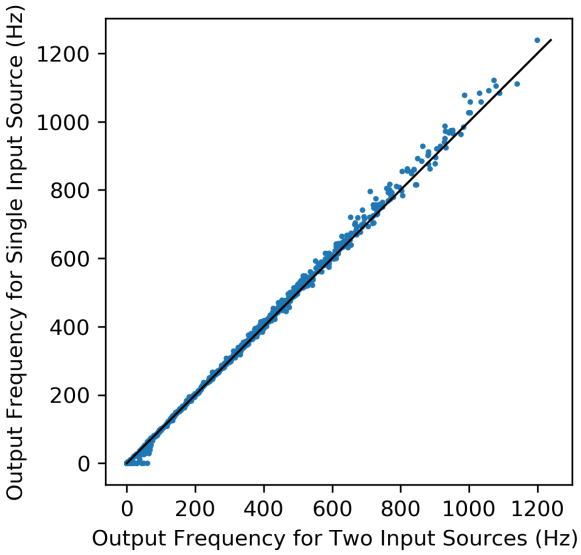


Figure 3.6: Validation results of our strategy for combining different inputs with different frequencies and weights. For our simulation, we used one output neuron and two Poisson input neurons whose average firing frequencies were drawn from a uniform distribution in the range of 0 to 5000 Hz. One synaptic weight of one of the two input sources was randomly drawn from a uniform distribution in the range of 0 to 0.2; the other weight was drawn from a uniform distribution in the range of -0.2 to 0.2. The latter connection can therefore be excitatory or inhibitory. The simulations were carried out for 5 seconds (of simulated time) and the output neuron's average output firing frequency was calculated.
In the second part of the experiment, the combined weight w' and frequency f' was calculated based on Eq. 3.33 and Eq. 3.34. On the simulational end, we used one Poisson input neuron whose average firing frequency is the combined frequency f' as well as another neuron which is receiving this input and is connected with the combined synaptic weight w'. The average output firing frequency was calculated in the same way as the first part.
The horizontal axis of the figure represents the output frequency of the neuron in the first part; the vertical axis represents the output frequency of the neuron in the second part (with the combined weights and frequencies). The results of 1000 trials are shown. The results reveal that the relation is strongly linear, close to the black line which indicates a perfect correspondence. All simulation were carried out using Brian 2 (Goodman and Brette, 2008).

3.3 Group dynamics

3.3.1 Analysis of 2 spiking neuron groups (1 Exc, 1 Inh)

After having introduced the mathematical tools we need for relating rate-based and spiking models, we now wish to turn to studying WTA behavior in spiking neuron groups.

As a first step, we here wish to start with a network containing only a single excitatory and a single inhibitory group. We define our input-output function as follows:

$$F(w,f) = \begin{cases} \frac{1}{t_{ref}-\tau \ln\left(1-\frac{\Delta v}{w}\left(1-\exp\left(-\frac{1}{f\tau}\right)\right)\right)} & f > f_\theta \\ 0 & f \leq f_\theta \end{cases} \tag{3.36}$$

$$f_\theta = -\frac{1}{\tau \ln\left(1-\frac{w}{\Delta v}\right)} \tag{3.37}$$

$$w \leq \Delta v \tag{3.38}$$

$$\Delta v = v_\theta - v_{rest} \tag{3.39}$$

Based on the combined weight and firing frequency policy described in the previous section, we can express the system in the following way:

$$\tau \frac{du_1}{dt} = -u_1 + F(w', f') \tag{3.40}$$

$$\tau \frac{du_2}{dt} = -u_2 + F(\beta_1, N_1 p_3 u_1) \tag{3.41}$$

$$w' = \frac{N_1 p_1 u_1}{f'}\alpha_1 - \frac{N_2 p_2 u_2}{f'}\beta_2 + \frac{f_{ext}}{f'}s \tag{3.42}$$

$$f' = N_1 p_1 u_1 + N_2 p_2 u_2 + f_{ext}, \tag{3.43}$$

where u_1 and u_2 are the average firing frequencies in the excitatory and inhibitory neuron groups. p_1, p_2, and p_3 are the probabilities for neuron connections within the excitatory group, connections from the inhibitory to the excitatory group, and connections from the excitatory to the inhibitory group, respectively. N_1 and N_2 are the number of neurons for the excitatory and the inhibitory group. f_{ext} represents the frequency of the external input (i.e. the input spikes), α_1 represents the synaptic weight for self excitation, and β_1 and $-\beta_2$ are the synaptic weights for connections from excitatory to inhibitory and from inhibitory to excitatory neurons, respectively.

3.3.1.1 Stability analysis based on rate-based results

Given what has been discussed above, we can apply the stability conditions of Rutishauser et al. (2011) following linearization as the Jacobian analysis is only concerned with the gradient of the system. To this end, we can linearize Eqs. 3.40 and 3.41 by Eq. 3.31:

$$\tau \frac{du_1}{dt} = -u_1 + \frac{w'}{\Delta v} f' + Const. \tag{3.44}$$

$$\tau \frac{du_2}{dt} = -u_2 + \frac{\beta_1}{\Delta v} N_1 p_2 u_1 + Const. \tag{3.45}$$

Please note that while the linearization is derived by setting f' and u_1 to infinity, the function's gradient does, in fact, converge rapidly. This is shown in Fig. 3.4. We are therefore able to use this equation even in case of medium-large finite values of f' and u_1. By making use of Eqs. 3.42 and 3.43, we can transform Eqs. 3.44 and 3.45 to:

$$\tau \frac{du_1}{dt} = -u_1 + \frac{\alpha_1}{\Delta v} N_1 p_1 u_1 - \frac{\beta_2}{\Delta v} N_2 p_2 u_2 + \frac{s}{\Delta v} f_{ext} + Const. \tag{3.46}$$

$$\tau \frac{du_2}{dt} = -u_2 + \frac{\beta_1}{\Delta v} N_1 p_3 u_1 + Const. \tag{3.47}$$

If we assume that both groups are active (they should be as we are assessing, by definition, the high-frequency regime), the Jacobian of this system, as detailed in the second chapter, is given by:

$$\tau \mathbf{J_2} = \begin{bmatrix} \frac{N_1 p_1 \alpha_1}{\Delta v} - 1 & -\frac{N_2 p_2 \beta_2}{\Delta v} \\ \frac{N_1 p_3 \beta_1}{\Delta v} & -1 \end{bmatrix} \tag{3.48}$$

When we define the following terms:

$$\alpha_1^* = \frac{N_1 p_1 \alpha_1}{\Delta v} \tag{3.49}$$

$$\beta_1^* = \frac{N_1 p_3 \beta_1}{\Delta v} \tag{3.50}$$

$$\beta_2^* = \frac{N_2 p_2 \beta_1}{\Delta v}, \tag{3.51}$$

we can rewrite Eq. 3.48 as follows:

$$\tau \mathbf{J_2^*} = \begin{bmatrix} \alpha_1^* - 1 & -\beta_2^* \\ \beta_1^* & -1 \end{bmatrix} \tag{3.52}$$

The condition for the system to be stable is that all real parts of the eigenvalues are negative. When solving for this, we end up with the following two conditions:

$$\alpha_1^* < 2 \cap \beta_1^* \beta_2^* > \alpha_1^* - 1 \tag{3.53}$$

These conditions are identical to those derived in the context of hard WTA. This is not surprising as, by definition, only two units are active in hard WTA.

We will now compare the analytically-derived conditions to results obtained from rate-based numerical approximations and spiking simulations. Details for both are provided later in this chapter. The left panel of Fig. 3.7 depicts the logarized ($\log_{10}$) firing frequency values of the numerically approximated fixed point analysis. The blank (white) boxes in the heatmap represent explosions in the numerical approximation. The darkly colored area does, in fact, match the theoretically determined stable area, defined by $\beta_1^* \beta_2^* > \alpha_1^* - 1$ and visually corresponding to the entire parameter space except for the bottom-right triangle. The right panel of Fig. 3.7 depicts simulated firing rates at external firing frequencies. Similar to the left panel, high values correspond to unstable behavior. While the spiking activity in the theoretically unstable area is not as extreme as results from the numerical approximation (left panel) indicate, both heatmaps are, globally, in keeping with the analytically derived conditions.

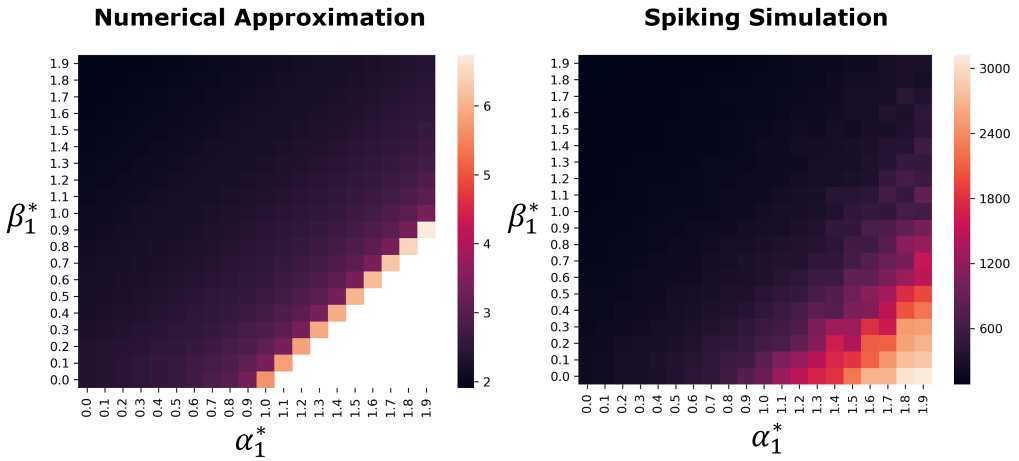


Figure 3.7: Stable parameter regions according to numerical approximation and spiking simulation for spiking network containing a single excitatory and a single inhibitory neuron group. Left panel: Depiction of logarized (log_{10}) firing frequency values. Right panel: Depiction of actual firing rates at external firing frequencies. The parameters for generating this figure are: $\beta_2 = 0.1$, $s = 0.05$, $\tau = 20$ms, $N_1 = N_2 = 100$, $\Delta v = 1$. The connection probability p between groups was globally set to 0.1. The firing frequency of external input was at 4750 (Hz) and the simulation was run for 500 ms. Note that $\beta_2^* = N_2 * p * \beta_2 = 1$. α_1 and β_1 were varied between 0 and 0.19 by steps of 0.01 (equivalent to α_1^* and β_1^* being varied between 0 and 1.9 by steps of 0.1)

3.3.1.2 Phase plane for activity prediction

Similar to our investigation in the previous chapter, we can carry out a phase plane analysis and predict the activity of units in our spiking model. The left panel of Fig. 3.8 depicts the phase plane for a network of one excitatory (u_1) and one inhibitory (u_2) neuron group: the red and blue line thereby represent the nullclines of the excitatory and inhibitory groups, respectively, and their intersections represent fixed points in the given parameter setting. The right panel of Fig. 3.8 depicts the units' activity that is predicted based on the fixed points in the phase plane. Note that, due to the difficulties of deriving fixed points analytically – particularly, when the number of units is large – we built a program to calculate a numerical approximation of the fixed points (see Appendix B).

To test the reliability of our approximation to the fixed points, we engaged in spiking simulations and compared the results to our numerical predictions. For the exemplary parameter set that was chosen for Fig. 3.8, such comparison is depicted in Fig. 3.9. This figure reveals that, for most input frequencies, the activity levels of our simulations are in agreement with our numerical predic-

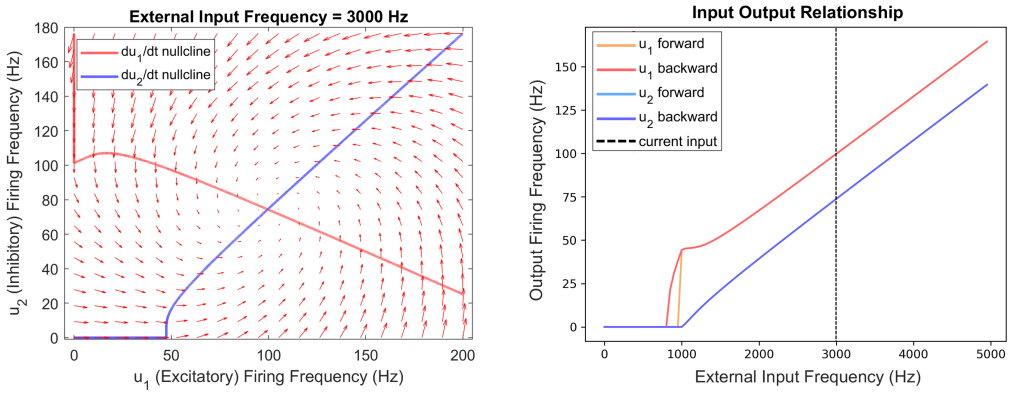


Figure 3.8: Depiction of phase plane and activity prediction. Left panel: Phase plane of one excitatory and one inhibitory group of LIF neurons. Right panel: Predicted activity of neuron groups. The dashed line in the right panel represents the external input based on which the phase plane in the left panel has been plotted. The parameters for generating this figure are: $\alpha_1 = 0.05$, $\beta_1 = \beta_2 = 0.1$, $s = 0.05$, $\tau = 20$ms, $N_1 = N_2 = 100$, $\Delta v = 1$, $v_{rest} = 0$. The connection probability p between groups is globally set to 0.1. The firing frequency of external input was varied between 0 and 5000 (Hz). The activity is calculated through the numerical approximation of the fixed point.

tions. Exceptions are low-frequency inputs. Here, theory and simulation produce different results. In particular, our theory predicted hysteresis behavior for input frequencies between 800 and 1000 Hz. This is evident from the difference between the forward (referring to gradually increasing external input) and backward (gradually decreasing input) lines for the first excitatory unit. In our simulations, however, these two curves are fully overlapping and no hysteresis can be observed.

While there is a difference between a small part of the low-frequency input regime, simulation and theory were, overall, found to match for the parameter setting used for generating Fig. 3.9. This match, however, is not always given. When increasing the self-excitation parameter α_1, the results from simulation and theory start to diverge. This is shown in Fig. 3.10. Note that as, in general, no hysteresis is observed in our spiking simulations, backward and forward curves are fully overlapping and subsequent plots will depict only forward versions of the simulations.

More systematically, the differences between results stemming from simulations and theory, for different values of α_1 and β_1, are depicted in Fig. 3.11. We see that when the self-excitation is large compared to the effect of the inhibition, this difference increases, especially in regions close to the stability limit.

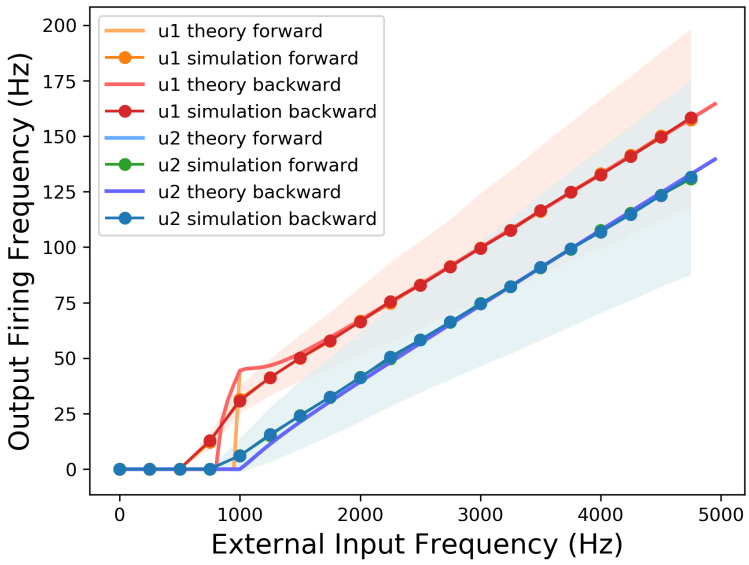


Figure 3.9: Activity prediction based on numerical approximation and simulation results compared. Forward curves refer to the behavior observed as a consequence of gradually increasing external input; backward curves to that of gradually decreasing input. The shaded area is representing ±SD. Simulation results are presented for the parameters used in Fig. 3.8. Parameters are given by: $\alpha_1 = 0.05$, $\beta_1 = \beta_2 = 0.1$, $s = 0.05$, $\tau = 20$ms. As for the simulations, for each point depicted, a simulation was run for 500 ms (simulated time). The external input frequency was varied from 0 to 4750 (Hz) in 20 steps. For the theory component, the numerical approximation was calculated using our numerical fixed point approximation. The external input was varied with 100 steps. Other parameters are provided by: $N_1 = N_2 = 100$, $p = 0.1$.

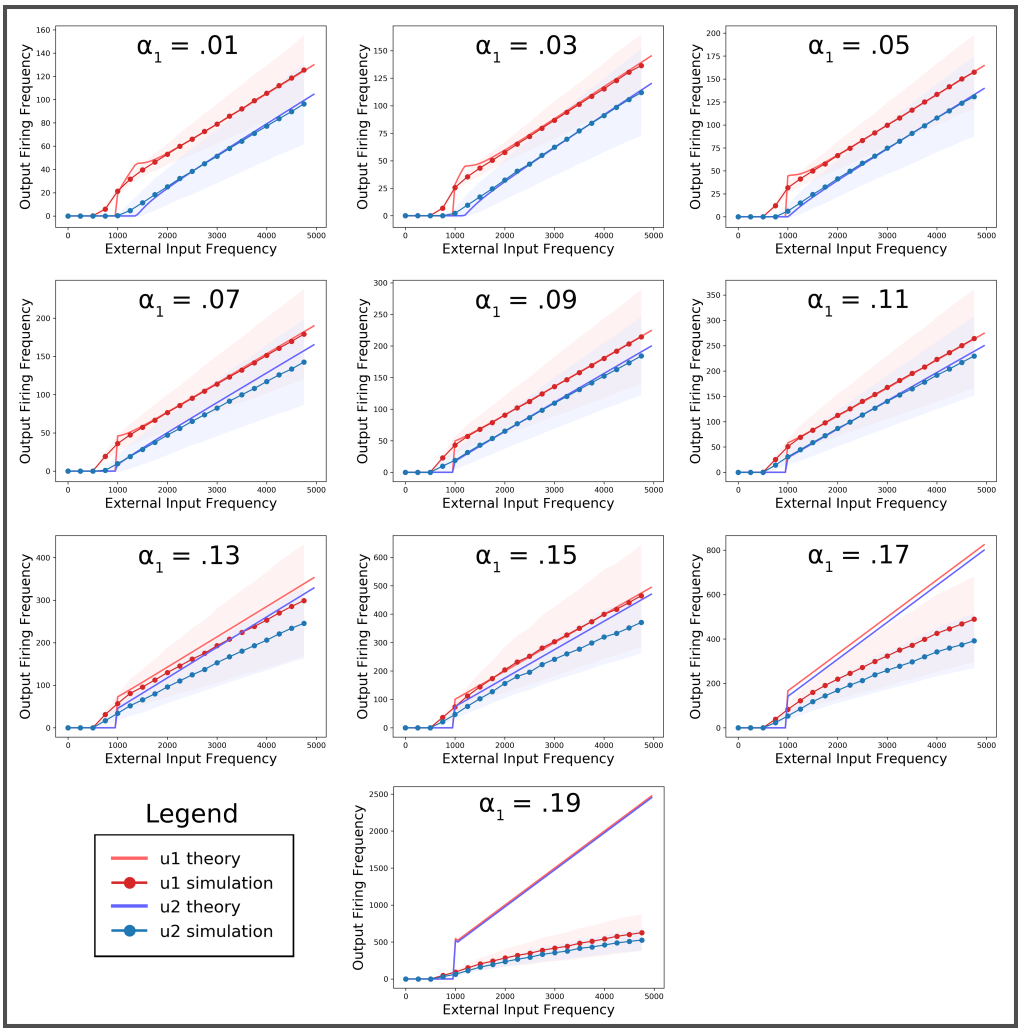


Figure 3.10: Depiction of the relationship between input and output firing frequencies in theory and simulation when varying the self-excitation parameter α_1. The figure shows that, as α_1 increases, theory and simulation differ more strongly. The simulation setting is the same as in Fig. 3.9 except for the varied α_1. All units are in Hz.

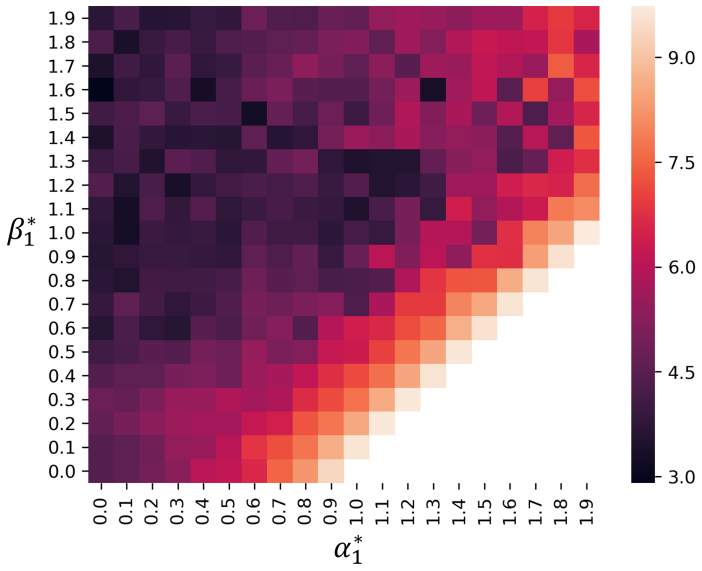


Figure 3.11: Depiction of differences between theoretical results and those stemming from simulations. These differences are calculated as the sum of absolute differences between theory and simulation for each simulated point along 20 different external input strengths.Therefore, the units are in Hz. These values are logarized ($\log_e$) and color-coded, with bright colors representing large differences and blank (white) boxes representing parameter sets that lead to unstable behavior. When the self-excitation is strong, compared to the inhibition, the difference between prediction and simulation becomes larger, especially in those regions that are close to the stability border. The simulation setting, again, is the same as in Fig. 3.9 except for the varied α_1 and β_1.

3.3.1.3 Effect of synchronization on activity prediction

After having established that the activity prediction tends to fail when the self-excitation, relative to the inhibition, is strong, we aimed to understand the mechanisms behind it. A visual inspection of activity raster plots led us to hypothesize that activity prediction might fail due to strong phase synchronization. We therefore decided to quantify the synchronization using the Phase Synchronization Index (PSI) as provided in Li et al. (2012) and originally presented in Rosenblum et al. (2001). In the PSI, full synchronization corresponds to an index of 1 while equally distributed phases in angle space correspond to 0. It is defined as follows:

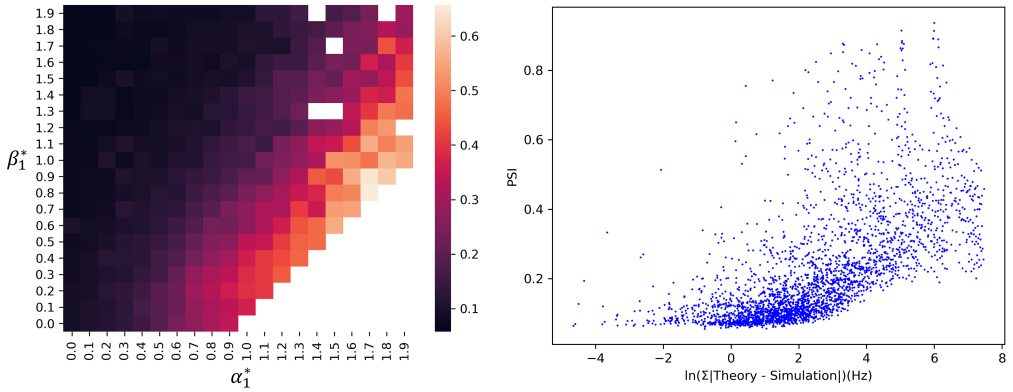
$$\mathrm{PSI}(t) = \frac{1}{N}\left|\sum_{j=1}^{N} \exp(i\phi_j(t))\right|, \tag{3.54}$$

where

$$\phi_j(t) = 2\pi k + 2\pi \frac{t - t_j^k}{t_j^{k+1} - t_j^k}, \ \left(t_j^k \leq t < t_j^{k+1}\right), \tag{3.55}$$

where $\phi_j(t)$ is the phase of neuron j at time t, t_j^k is the timing of the k^{th} spike of neuron j, and i is the imaginary unit ($i = \sqrt{-1}$).

Fig. 3.12a depicts the mean PSI values in different areas of the parameter space. Blank (white) entries thereby represent either an unstable area or a setting where PSI could not be calculated (i.e., when one or more neurons in the group did not spike once throughout the course of the entire simulation). This figure reveals that when the self-excitation is strong and the inhibition is rather weak, stronger synchronization is observed. This is related to the deviation between theory and simulation, as is shown in Fig. 3.12b.



(a) Mean PSI values (b) PSI relation to error in prediction

Figure 3.12: Relation between the phase synchronization index (PSI) and difference between theory and prediction depicted. Left panel: Mean PSI values for the excitatory neuron group in different parameter regions. The mean PSI was calculated by averaging the PSI over the 20 simulations with different external input strengths. Right panel: Non-random relation between PSI and difference between theory and simulation visualized. In this plot, instead of the means, all data points, except those where the PSI could not be calculated, are depicted. The simulation setting, again, is the same as in Fig. 3.9 except for the varied α_1 and β_1.

3.3.1.4 Connection probability and phase synchronization

As a side investigation, we wished to assess the effect of the connection probability p on the phase synchronization observed. We therefore engaged in the following simulations: we varied α_1^* and β_1^* across a range of values and set the connection probability, p, to 0.1, 0.5, and 0.9, respectively. For each combination of parameters, we simulated the time course for 500 ms (simulated time) when applying an external input frequency of 5000 Hz. The PSI was assessed and plotted for all combinations, except for when the area was unstable or when the PSI could not be calculated. The results are shown in Fig. 3.13. They reveal that for higher connection probabilities, the mean and variance are (slightly) increased.

Fig. 3.14 depicts the results of a second assessment. Here, for each parameter set, the external input frequency was changed from 250 Hz to 5000 Hz in steps of 250. Each configuration was simulated for a total duration of 500 ms (simulated time). For the same set of parameters, theoretical firing frequencies were calculated and absolute differences between theory and simulation summed along all different external stimulus levels. These values were logarized to facilitate visual inspection of the heatmap. Comparing the results of Fig. 3.14 to Fig. 3.13, we see that strongly synchronized areas are associated with large deviations between theory and simulation. The scatter plot also reveals that when the PSI is high, the difference between theory and simulation tends to be large.

Interestingly, even in case of high connection probabilities, certain parameter regions show low synchronization. Choosing these parameters in that context will lead to predictable behavior of the system (e.g. when α is low and β high). Moreover, note that for a connection probability of 0.1, when there is a large gap between simulation and theory, the activity level is overestimated by the theoretical prediction, whereas for $p = 0.5$ and $p = 0.9$, the activity was underestimated as well. In both cases, synchronization leads to a reduction of the precision of the prediction function.

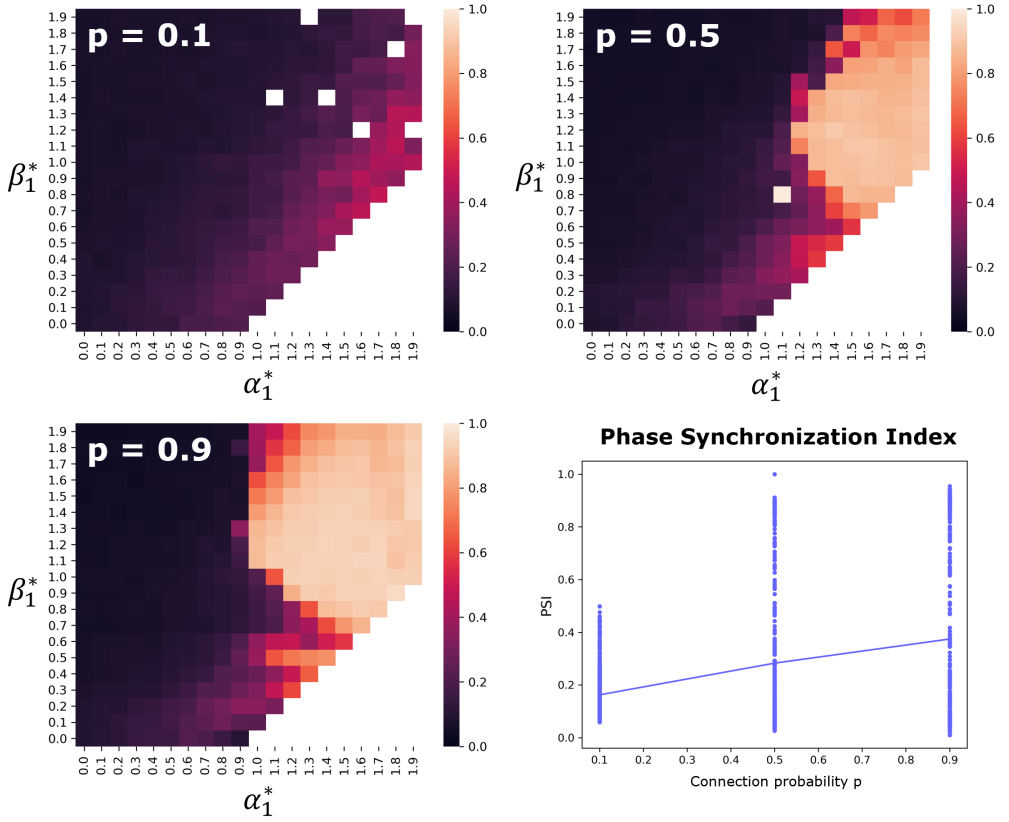


Figure 3.13: PSI as a function of connection probability. For a wide range of parameters and as a function of the connection probability p, the PSI was calculated and plotted as a heatmap. Blank (white) boxes indicate parameter settings refer to either unstable behavior (see Fig. 3.7) or to a scenario where the PSI could not be calculated (i.e., one or more neurons did not fire throughout the simulation period). Note that α_1^* and β_1^* β_2^* include the connection probability and group size: $\alpha_1^* = N_1 p_1 \alpha_1$. Therefore, we adapted the actual α_1 and β_1 β_2 such that all the α_1^* and β_1^* are the same. In all simulations, $\beta_1^* = 1$, $N_1 = N_2 = 100$, and $\Delta v = 1$ holds. The scatter plot in the bottom-right corner depicts the relation between connection probability and PSI. All points represent the PSI values in the heatmap; the line represents their mean. We see that the mean is loosely increasing and that for connection probabilities of 0.5 and 0.9, the variance is further increased.

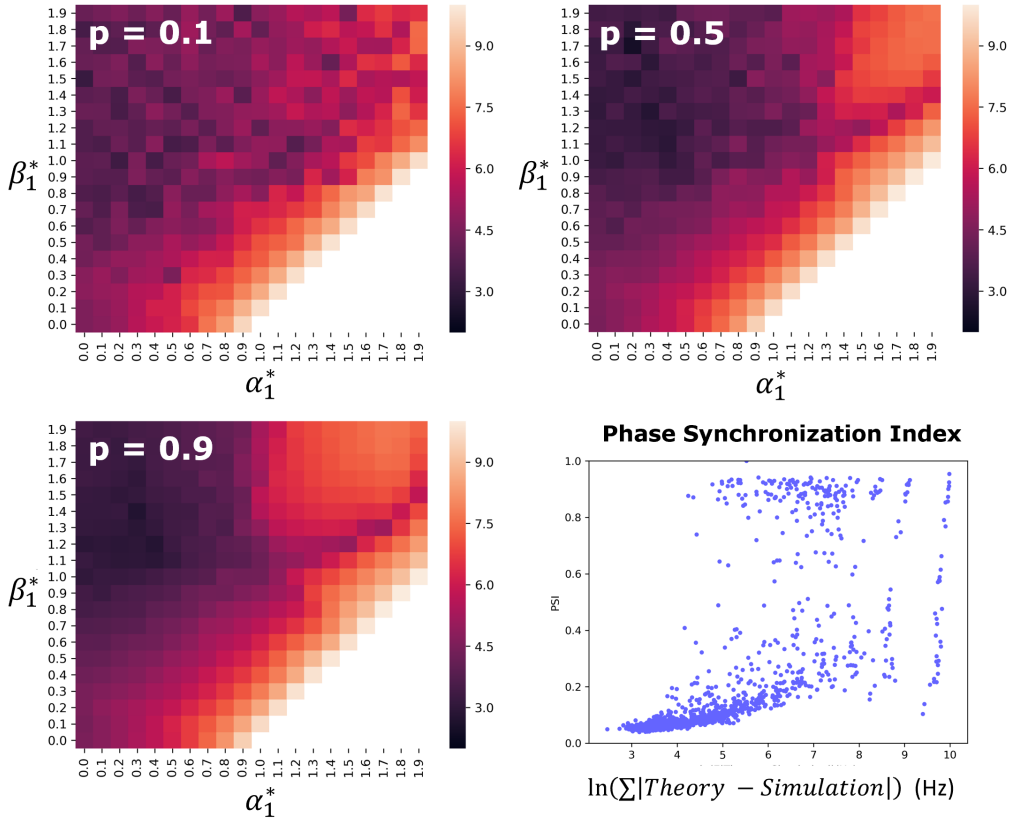


Figure 3.14: Effect of connectivity on difference between theory and simulations. The top-left, top-right, and bottom-left panels depict heatmaps of the sum of absolute differences between simulation and theory, logarized ($\log_e$) for a given connection probability. The bottom-right panel depicts the relation between the phase synchronization index and the logarized error between theory and simulation. In our simulations, the external input frequency was varied from 250 Hz to 5000 Hz in 20 steps. The other simulation settings identical with those presented in Fig. 3.13.

3.3.2 WTA with 3 spiking neuron groups (2 Exc, 1 Inh)

A network with three groups, as depicted in Fig. 2.1 A (without inter-excitatory connections), can be formalized through the following differential equations:

$$\tau \frac{du_1}{dt} = -u_1 + F(w', f') \tag{3.56}$$
$$\tau \frac{du_2}{dt} = -u_2 + F(w'', f'') \tag{3.57}$$
$$\tau \frac{du_3}{dt} = -u_3 + F(\beta_1, Npu_1 + Npu_2) \tag{3.58}$$
$$w' = \frac{Npu_1}{f'}\alpha_1 - \frac{Npu_3}{f'}\beta_2 + \frac{f_{ext_1}}{f'}s \tag{3.59}$$
$$f' = Npu_1 + Npu_3 + f_{ext_1} \tag{3.60}$$
$$w'' = \frac{Npu_2}{f''}\alpha_1 - \frac{Npu_3}{f''}\beta_2 + \frac{f_{ext_2}}{f''}s \tag{3.61}$$
$$f'' = Npu_2 + Npu_3 + f_{ext_2}, \tag{3.62}$$
$$\tag{3.63}$$

where N is the number of neurons in each group (setting a common number for simplicity) and p is the connection probability between groups (set to be identical, for simplicity).

Similar to the case of two groups, we can apply linearization:

$$\tau \frac{du_1}{dt} = -u_1 + \frac{\alpha_1}{\Delta v}Npu_1 - \frac{\beta_2}{\Delta v}Npu_3 + \frac{s}{\Delta v}f_{ext_1} + C \tag{3.64}$$
$$\tau \frac{du_2}{dt} = -u_2 + \frac{\alpha_1}{\Delta v}Npu_2 - \frac{\beta_2}{\Delta v}Npu_3 + \frac{s}{\Delta v}f_{ext_2} + C \tag{3.65}$$
$$\tau \frac{du_3}{dt} = -u_3 + \frac{\beta_1}{\Delta v}(Npu_1 + Npu_2) + C, \tag{3.66}$$

where C is a constant (different in each usage, it is just *some* constant).

Now, if we define the following variables:

$$\alpha_1^* = \frac{Np\alpha_1}{\Delta v} \tag{3.67}$$
$$\beta_1^* = \frac{Np\beta_1}{\Delta v} \tag{3.68}$$
$$\beta_2^* = \frac{Np\beta_2}{\Delta v}, \tag{3.69}$$

we can obtain the Jacobian for the system with dummy variables:

$$\tau \mathbf{J}_{\mathbf{2}}^{*} = \begin{bmatrix} l_1\alpha_1^* - 1 & 0 & -l_3\beta_2^* \\ 0 & l_2\alpha_1^* - 1 & -l_3\beta_2^* \\ l_1\beta_1^* & l_2\beta_1^* & -1 \end{bmatrix} \tag{3.70}$$

This, essentially, yields the same conditions as those presented in Fig. 3.13.

Hard WTA:

$$\alpha_1^* < 2 \tag{3.71}$$

$$\beta_1^*\beta_2^* > \alpha_1^* - 1 \tag{3.72}$$

Soft WTA:

$$\alpha_1^* < 1 \tag{3.73}$$

These conditions we were able to illuminate and validate through spiking simulations.

First, to provide a visual intuition, Fig. 3.15 shows the time course of the units' activities for soft WTA (see Fig. 3.15, panel 1) and hard WTA (see Fig. 3.15, panel 2). Panel A of Fig. 3.15 further depicts the parameter range that results in stable soft or hard WTA behavior.

Second, similar to Fig. 3.7, the left panel of Fig. 3.16 depicts the logarized firing frequency values of the numerically approximated fixed points while the right panel shows the actual firing rates at the external firing frequency. Similar to the network consisting of two groups, because the stable conditions are identical in the end, while the spiking results in the unstable area are not as extreme as the results from the numerical approximation would indicate, both plots are generally in agreement with the analytically derived conditions.

Third, Fig. 3.17 further illustrates the stable soft and hard WTA parameter regions in our spiking network, according to both numerical approximation and spiking simulation. The color of the heatmaps indicates the fraction of the winning group's average firing rates relative to the sum of firing rates of all groups. The results are in keeping with our expectation from Fig. 3.15: the area on the right side of the blue line in Fig. 3.17 represents the hard WTA region while the parameter space on the left of it is associated with soft or hard WTA. We see that the ratio of winners in the hard WTA region is markedly high. In the soft or hard-WTA area, we see a similarly high profile for parts of the heatmap and reduced activity for other parts. This is expected as the left area includes both soft and hard WTA.

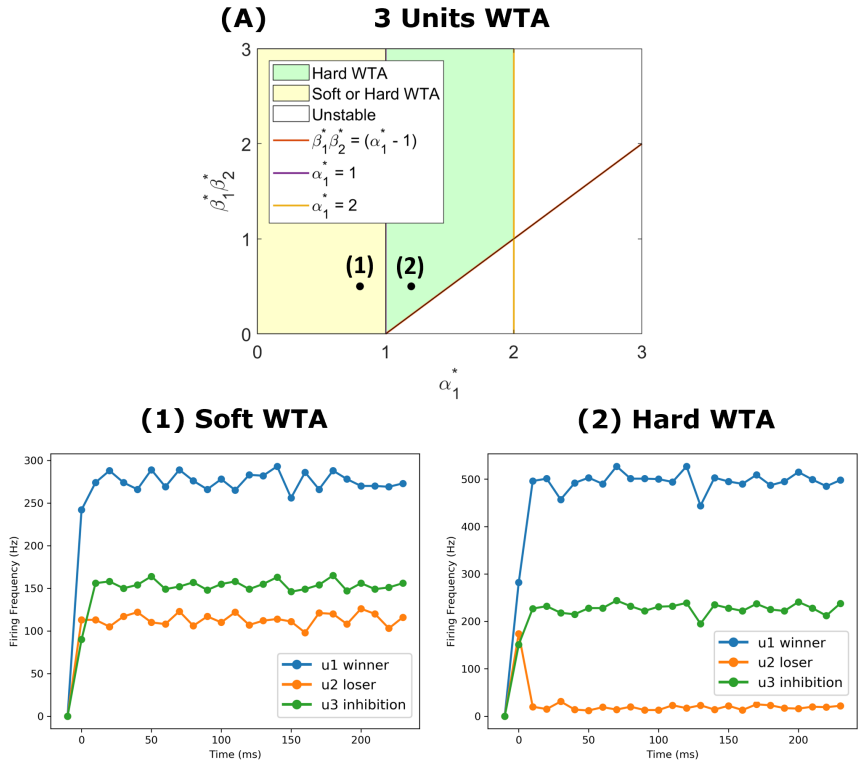


Figure 3.15: Spiking time course for soft and hard WTA networks with two excitatory neuron groups and a single inhibitory group upon receiving a 5000 Hz input with a 0.05 synaptic weight for the winner and a 0.04 weight for the loser. Panels 1 and 2 depict the units' activity in soft and hard WTA configurations, respectively. Panel A depicts the theoretical range of parameters that lead to stable hard and soft WTA behavior. Parameters in this simulations are $\beta_1 = 0.1$, $\beta_2 = 0.05$, $\alpha_1 = 0.08$ for panel 1, and $\alpha_1 = 0.12$ for panel 2. The external input in the plots starts at 0 ms. Other parameters are given by: $N = 100$, $p = 0.1$, $\Delta v = 1$.

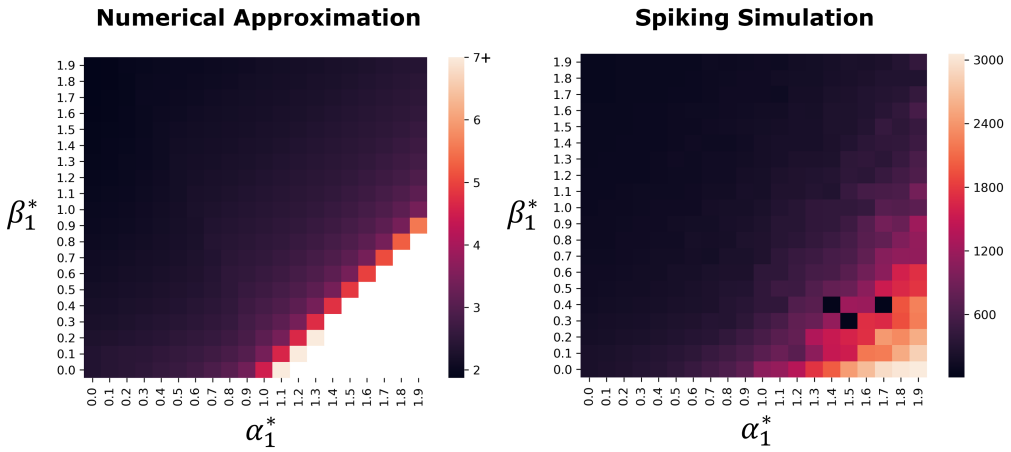


Figure 3.16: Stable parameter regions according to numerical approximation and spiking simulation for spiking network containing 2 excitatory and 1 inhibitory neuron groups. Left panel: Logarized ($\log_{10}$) firing frequency value based on fixed point. The dark colored area matches the theoretically stable area, as in Rutishauser et al. (2011): $\beta_1^*\beta_2^* > \alpha_1^* - 1$. Right panel: Average firing rates of the winner unit at the external firing frequency at 5000 Hz. Parameters are the same as those used in Fig. 3.15. Note that a few completely black boxes are visible in the otherwise exploding area. A follow-up investigation revealed that, in these configurations, there was a supposed-to-be-loser unit that ended up being the winner. This is due the noise element in the simulations.

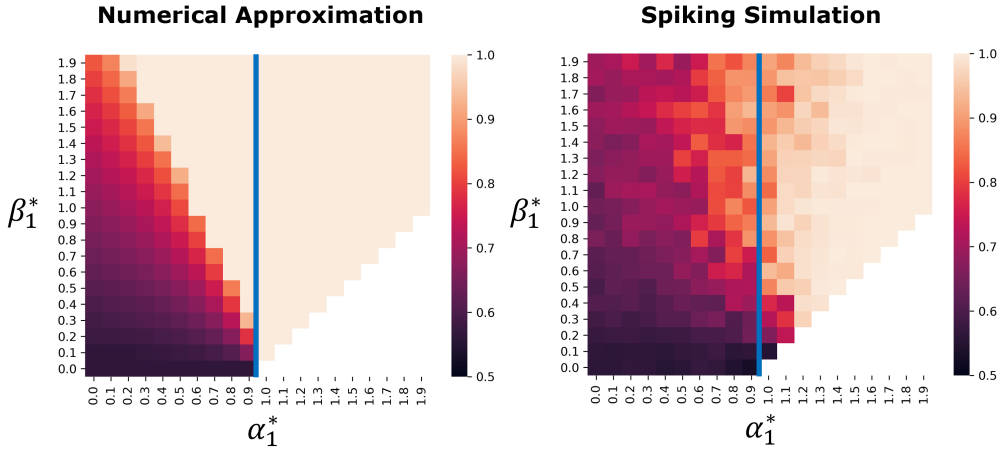


Figure 3.17: Illustration of stable soft and hard WTA parameter region in spiking neural network with 3 neuron groups (2 excitatory, 1 inhibitory) according to numerical approximations and spiking simulations. The network structure is the same as the one shown in Fig. 2.1 A. One excitatory unit receives input with 80 % of the strength of the other unit. Simulations were carried out for 1 second and the average firing frequency ratio was calculated based on the activity in the latter 500ms of this interval. The x-axis of both panels represents the effective self-excitation value, i.e., $\alpha_1^* = \alpha_1 Np$, where N is the number of neurons in the group and p is the connection probability. Likewise, the y-axis is given in $\beta_1^* = \beta_1 Np$. The color of the heatmaps indicates the fraction of the winning group's average firing rates relative to the sum of firing rates of all groups. For reference, without any interaction or inhibition, the ratio would be $\frac{1}{1+0.8} = 0.556$. Other parameters in this simulation are set as follows: $\beta_2^* = 1$, $\Delta v = 1$. The external input spike frequency (Poisson) is given by 5000 Hz.

3.3.3 WTA with 4 spiking neuron groups (3 Exc, 1 Inh)

Here, we wish to carry out a simulation for a 4-unit network of neuron groups arranged as in network B' in Fig. 2.7 which is equipped with inter-excitatory connections α_2. The system can be described through the following differential equations:

$$\begin{aligned}
\tau\frac{du_1}{dt} &= -u_1 + F(w', f') && (3.74)\\
\tau\frac{du_2}{dt} &= -u_2 + F(w'', f'') && (3.75)\\
\tau\frac{du_3}{dt} &= -u_3 + F(w''', f''') && (3.76)\\
\tau\frac{du_4}{dt} &= -u_4 + F(\beta_1, Npu_1 + Npu_2 + Npu_3) && (3.77)\\
w' &= \frac{Npu_1}{f'}\alpha_1 + \frac{Np(u_2+u_3)}{f'}\alpha_2 - \frac{Npu_4}{f'}\beta_2 + \frac{f_{ext_1}}{f'}s && (3.78)\\
f' &= Np(u_1+u_2+u_3) + f_{ext_1} && (3.79)\\
w'' &= \frac{Npu_2}{f''}\alpha_1 + \frac{Np(u_1+u_3)}{f''}\alpha_2 - \frac{Npu_4}{f''}\beta_2 + \frac{f_{ext_2}}{f''}s && (3.80)\\
f'' &= Np(u_1+u_2+u_3) + f_{ext_2} && (3.81)\\
w''' &= \frac{Npu_3}{f'''}\alpha_1 + \frac{Np(u_1+u_2)}{f'''}\alpha_2 - \frac{Npu_4}{f''}\beta_2 + \frac{f_{ext_2}}{f''}s && (3.82)\\
f''' &= Np(u_1+u_2+u_3) + f_{ext_3}, && (3.83)\\
& && (3.84)
\end{aligned}$$

where N is the number of neurons in each group and p the connection probability between groups.

Similar to the 3-unit instance, we can apply linearization:

$$\tau\frac{du_1}{dt} = -u_1 + \frac{Np\alpha_1}{\Delta v}u_1 + \frac{Np\alpha_2}{\Delta v}(u_2+u_3) - \frac{Np\beta_2}{\Delta v}u_4 + \frac{s}{\Delta v}f_{ext_1} + C \quad (3.85)$$

$$\tau\frac{du_2}{dt} = -u_2 + \frac{Np\alpha_1}{\Delta v}u_2 + \frac{Np\alpha_2}{\Delta v}(u_1+u_3) - \frac{Np\beta_2}{\Delta v}u_4 + \frac{s}{\Delta v}f_{ext_2} + C \quad (3.86)$$

$$\tau\frac{du_3}{dt} = -u_3 + \frac{Np\alpha_1}{\Delta v}u_3 + \frac{Np\alpha_2}{\Delta v}(u_1+u_2) - \frac{Np\beta_2}{\Delta v}u_4 + \frac{s}{\Delta v}f_{ext_3} + C \quad (3.87)$$

$$\tau\frac{du_4}{dt} = -u_4 + \frac{Np\beta_1}{\Delta v}(u_1+u_2+u_3) + C. \quad (3.88)$$

Next, we define the following variables:

$$\alpha_1^* = \frac{Np\alpha_1}{\Delta v} \tag{3.89}$$

$$\alpha_2^* = \frac{Np\alpha_2}{\Delta v} \tag{3.90}$$

$$\beta_1^* = \frac{Np\beta_1}{\Delta v} \tag{3.91}$$

$$\beta_2^* = \frac{Np\beta_2}{\Delta v}, \tag{3.92}$$

The Jacobian for this system, with dummy variables, can be derived as follows:

$$\tau \mathbf{J_3^*} = \begin{bmatrix} l_1\alpha_1^* - 1 & l_2\alpha_2^* & l_3\alpha_2^* & -l_4\beta_2^* \\ l_1\alpha_2^* & l_2\alpha_1^* - 1 & l_3\alpha_2^* & -l_4\beta_2^* \\ l_1\alpha_2^* & l_2\alpha_2^* & l_3\alpha_1^* - 1 & -l_4\beta_2^* \\ l_1\beta_1^* & l_2\beta_1^* & l_3\beta_1^* & -1 \end{bmatrix} \tag{3.93}$$

This is equivalent to Eq. 2.40. Therefore, we can simply use the border conditions for the hard/soft WTA and stable/unstable areas presented in Section 2.3.2. The conditions can be summarized as follows:

Stable hard WTA conditions:

$$\alpha_1^* < 2 \tag{3.94}$$

$$\beta_1^*\beta_2^* > \alpha_1^* - 1 \tag{3.95}$$

Stable soft WTA conditions:

$$\alpha_1^* - \alpha_2^* < 1 \tag{3.96}$$

$$\alpha_1^* + 2\alpha_2^* < 2 \tag{3.97}$$

$$\beta_1^*\beta_2^* > \frac{1}{3}(\alpha_1^* + 2\alpha_2^* - 1) \tag{3.98}$$

In order to produce figures similar to those in the previous sections, we set $\alpha_2^* = 0.1$. Fig. 3.18 shows the theoretically derived stable parameter regions with $\alpha_2^* = 0.1$. Fig. 3.19 depicts stable/unstable parameter regions, and Fig. 3.20 depicts soft/hard WTA parameter regions obtained through both numerical approximation and spiking simulation. They are, roughly, in agreement with the analytical conditions.

To conclude this chapter, we have seen for 3 spiking groups without inter-excitatory connections and for 4 spiking groups with inter-excitatory connections that the analytical work carried out in the context of the rate-based model by Rutishauser et al. (2011) can be translated to spiking neural networks. This approach can easily be extended to larger networks.

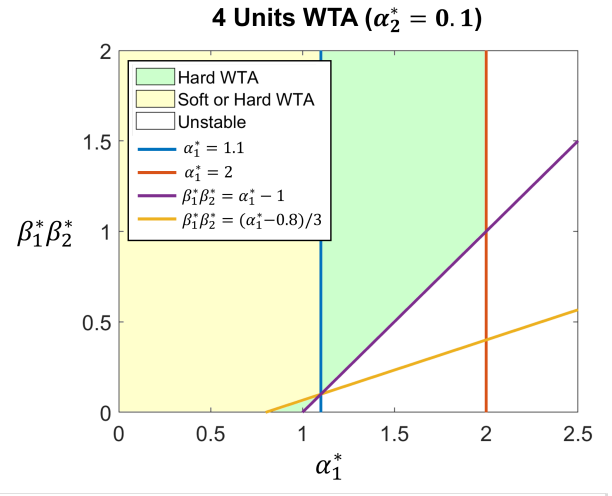


Figure 3.18: Stability in 4-unit WTA with $\alpha_2^* = 0.1$. Analytically derived soft and hard regions are provided.

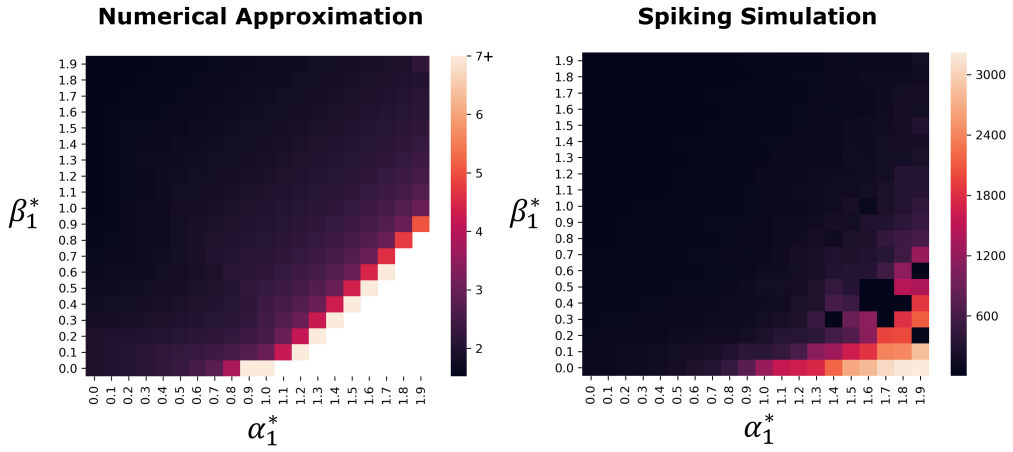


Figure 3.19: Stable parameter regions according to numerical approximation and spiking simulation for spiking network containing 3 excitatory and 1 inhibitory neuron group. The plot and analysis method is adapted from Fig. 3.16. It is worth noting that there is a small region in the theory (Fig. 3.18) where α_1^* is smaller than the soft/hard WTA border but only stable with hard WTA (the green triangle around $\alpha_1^* = 1$ and $\beta_1^*\beta_2^* = 0$). In the numerical simulations, the parameter sets that fell into this region were only those without any inhibition ($\beta_1^* = 0$) and thus showed unstable behavior as they cannot possibly take the hard WTA state.

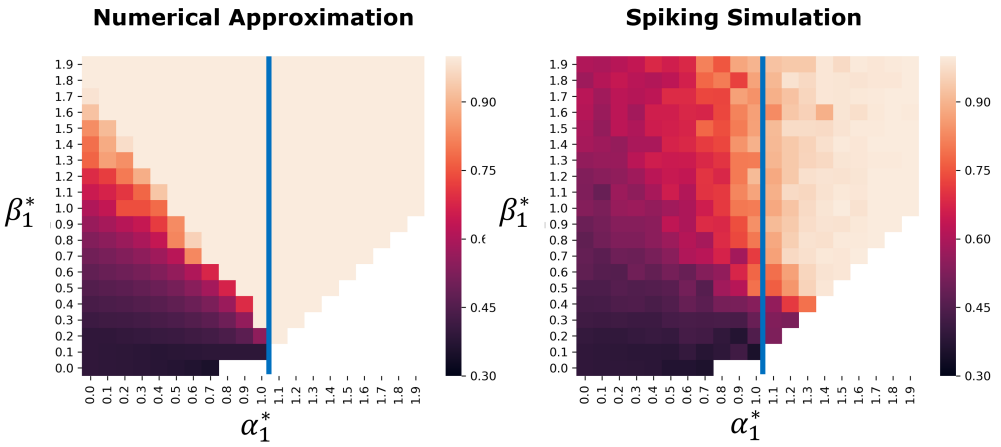


Figure 3.20: Illustration of stable soft and hard WTA parameter region in spiking neural network with 4 neuron groups (3 excitatory, 1 inhibitory) according to numerical approximations and spiking simulations. The plot and analysis is adapted from Fig. 3.17. For reference, without any interaction or inhibition, the ratio would be $\frac{1}{1+0.8+0.8} = 0.384$. Other parameters in this simulation are given by: $\beta_2 = 1$, $\Delta v = 1$. The external Poisson input spike frequency (Poisson) is 5000 Hz.

Chapter 4

Discussion

Here, we will review and discuss the main findings of this thesis.

In the second chapter, we focussed on a stability assessment of rate-based neuron models implementing winner-take-all dynamics. As a first step, we thereby reproduced and slightly modified the work of Rutishauser et al. (2011) who propose a formalism for neuronal activity within simple WTA networks and present an analytical approach for assessing their stability. While we were generally able to reproduce their results, we noticed two technical differences that we consider worth discussing.

First, while we were able to derive the same set of stability conditions using both a Hermitian and Jacobian approach, in the discussion of their paper, Rutishauser et al. (2011) state that they could not have succeeded in deriving analytical conditions using the Jacobian methods as they *"rely on linearizations and do not provide global stability conditions"* (Rutishauser et al., 2011). This is true when the system is non-linear. Indeed, with the activation function $max(0, x)$, their model does include a non-linearity. However, since they linearize the system by introducing the dummy variables l_i, the global conditions would, in fact, have been derivable with the Jacobian method. Second, Rutishauser et al. (2011) derive slightly stronger conditions for soft-WTA stability in a 3-unit network as they appear to be neglecting a particular parameter region ($\alpha_1^2 - 4\beta_1\beta_2 < 0$). As a consequence, they note that their analytical solution assigns an upper bound to the parameter β_2, which is, in fact, not existent in their simulations. By carrying out a careful case differentiation, our approach results in no such discrepancy.

Next, we extended the analyses that were originally carried out on a 3-unit WTA network to networks with an arbitrary number of excitatory units and a single inhibitory unit. Specifically, by mathematical induction, we derived a general form of the eigenvalues of the Jacobian of the network and concluded a series of parameter conditions for hard and soft WTA. This extension could prove useful when aiming to analyze larger networks. Notably, one of the key advantages of a rate-based model is its scalability. In the context of dynamic field theory, where dynamic neural fields assume a continuous representation, a generalization to arbitrarily many units appears imperative. While we set out

to do so, we have to acknowledge that our analysis assumes a specific type of connectivity: we assume that all connections between excitatory unit pairs are given by the same weight. In certain contexts, a more complex structure might be more desirable. Deriving conditions for more general connectivity patterns will therefore be a crucial component of future investigations.

In addition to stability conditions in hard and soft WTA regimes, we derived conditions for hysteresis and self-sustained behavior. This type of assessment could prove useful to more carefully study the dynamics of spiking networks, with self-sustained behavior furthermore touching upon an important theme in the context of working memory and dynamic field theory. A limitation we have to acknowledge is that the derivation of conditions for hysteresis and self-sustained behavior is, thus far, restricted to networks containing a single excitatory and a single inhibitory unit. While this analysis generalizes to hysteresis in large networks with hard WTA configurations (the hard WTA conditions are irrespective of the number of excitatory units as only one winner is selected), the hysteresis conditions for large soft WTA networks are less clear. We will therefore look into a broader characterization in the future.

In the third chapter, we were aiming to relate the parameter ranges in which the rate-based networks were found to exhibit desired behaviors to parameters in a spiking model. To this end, we derived a function that would capture the input-output firing rate relationship for a single leaky-integrate-and-fire neuron when assuming regular spiking input and utilized a strategy resembling the mean-field approach to generate a spiking neural network and map its parameters to the rate-based ones. While we generally succeeded in doing so, we here wish to highlight that this derivation itself is not the key achievement of this thesis – the mean-field approach is well established and has been applied numerous times to LIF neurons before. Instead, it only serves as a means to relate the separate rate-based findings from the second chapter to our spiking neuron models and to, subsequently, explore the parameter space for winner-take-all dynamics. Having had specific WTA dynamics and assumptions in mind, we were able to directly approach a subproblem. As a result, we provide an easy-to-follow derivation scheme.

Crucially, using this scheme, we were able to map the borders between parameter ranges that separate some of the different dynamical winner-take-all behaviors that we explored in the previous step. Specifically, we saw that for 3 and 4 spiking groups with and without inter-excitatory connections, the analytical work carried out in the context of Rutishauser et al. (2011) can be translated to spiking neural networks. Based on the analytical derivation for (n+1)-unit networks, this approach can be extended to various networks, if computational resources suffice. While we succeeded in mapping soft and hard WTA behavior, we have to acknowledge that we were not yet able to do so for hysteresis and self-sustained behavior. This will be the subject of further investigations.

Further, based on a number of factors – the input-output relationship, the system's phase planes, and numerical approximations of fixed points on those phase planes – we provided a firing rate prediction. When comparing this prediction to the spiking network simulations, we found that the prediction is accurate when neurons of the spiking model are weakly connected. For strong connectivity, however, it becomes inaccurate. This confirms previously discussed limitations of mean-field approaches. In this context, we also discuss a potential explanation for the poor activity prediction in certain parameter settings: phase synchronization.

In addition to the points already outlined above, several follow-ups appear plausible. For instance, one could consider carrying out a more general derivation of the eigenvalues and resulting WTA conditions in the model by Rutishauser et al. (2011), assuming that the neighbor connection strengths are following Gaussian distributions rather than assuming they adhere to the strict network structure we described. Further, one could build a direct connection to the model by Amari (1977) by introducing sigmoidal activation functions. Given that in the presence of a refractory period the activation function saturates at some point, this extension could prove to be a very fruitful one. Finally, one could relate the work presented here more directly to that of Wilson and Cowan (1973).

Overall, the pipeline presented here could prove useful in assisting the tuning of spiking neural network parameters to achieve desired behaviors in the WTA-framework, in particular on neuromorphic hardware. This is also promising in the context of dynamic neural fields that, under certain constraints, are equivalent to soft winner-take-all networks and represent *"a step toward cognitive neuromorphic architectures"* (Sandamirskaya, 2014).

Bibliography

Adrian, E. D. and Zotterman, Y. (1926). The impulses produced by sensory nerve-endings: Part ii. the response of a single end-organ. *The Journal of physiology*, 61(2):151–171.

Amari, S.-i. (1977). Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological cybernetics*, 27(2):77–87.

Beurle, R. L. (1956). Properties of a mass of cells capable of regenerating pulses. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, pages 55–94.

Binas, J., Rutishauser, U., Indiveri, G., and Pfeiffer, M. (2014). Learning and stabilization of winner-take-all dynamics through interacting excitatory and inhibitory plasticity. *Frontiers in computational neuroscience*, 8:68.

Brunel, N. (2000). Dynamics of sparsely connected networks of excitatory and inhibitory spiking neurons. *Journal of computational neuroscience*, 8(3):183–208.

Coombes, S. (2006). Neural fields. *Scholarpedia*, 1(6):1373.

Creutzfeldt, O. D. (1977). Generality of the functional structure of the neocortex. *Naturwissenschaften*, 64(10):507–517.

Douglas, R. J., Koch, C., Mahowald, M., Martin, K., and Suarez, H. H. (1995). Recurrent excitation in neocortical circuits. *Science*, 269(5226):981–985.

Douglas, R. J. and Martin, K. A. (2007). Recurrent neuronal circuits in the neocortex. *Current biology*, 17(13):R496–R500.

Douglas, R. J., Martin, K. A., and Whitteridge, D. (1989). A canonical microcircuit for neocortex. *Neural computation*, 1(4):480–488.

Fang, Y., Cohen, M. A., and Kincaid, T. G. (1996). Dynamics of a winner-take-all neural network. *Neural Networks*, 9(7):1141–1154.

Feldman, J. A. and Ballard, D. H. (1982). Connectionist models and their properties. *Cognitive science*, 6(3):205–254.

Gerstner, W. (2000). Population dynamics of spiking neurons: fast transients, asynchronous states, and locking. *Neural computation*, 12(1):43–89.

Gerstner, W. and Kistler, W. M. (2002). *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press.

Gerstner, W., Kistler, W. M., Naud, R., and Paninski, L. (2014). *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press.

Goodman, D. F. and Brette, R. (2008). Brian: a simulator for spiking neural networks in python. *Frontiers in neuroinformatics*, 2:5.

Heeger, D. (2000). Poisson model of spike generation. *Handout, University of Standford*, 5:1–13.

Herz, A. V., Gollisch, T., Machens, C. K., and Jaeger, D. (2006). Modeling single-neuron dynamics and computations: a balance of detail and abstraction. *science*, 314(5796):80–85.

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

Itti, L., Koch, C., and Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (11):1254–1259.

Izhikevich, E. M. (2007). Equilibrium. *Scholarpedia*, 2(10):2014.

Kutta, W. (1901). Beitrag zur naherungsweisen integration totaler differentialgleichungen. *Z. Math. Phys.*, 46:435–453.

Lazzaro, J., Ryckebusch, S., Mahowald, M. A., and Mead, C. A. (1989). Winner-take-all networks of o (n) complexity. In *Advances in neural information processing systems*, pages 703–711.

Li, J., Katori, Y., and Kohno, T. (2012). An fpga-based silicon neuronal network with selectable excitability silicon neurons. *Frontiers in Neuroscience*, 6:183.

Maass, W. (1997). Networks of spiking neurons: the third generation of neural network models. *Neural networks*, 10(9):1659–1671.

Maass, W. (2000). On the computational power of winner-take-all. *Neural computation*, 12(11):2519–2535.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133.

Montemurro, M. A., Rasch, M. J., Murayama, Y., Logothetis, N. K., and Panzeri, S. (2008). Phase-of-firing coding of natural visual stimuli in primary visual cortex. *Current biology*, 18(5):375–380.

Riesenhuber, M. and Poggio, T. (1999). Hierarchical models of object recognition in cortex. *Nature neuroscience*, 2(11):1019.

Rosenblum, M., Pikovsky, A., Kurths, J., Schäfer, C., and Tass, P. A. (2001). Phase synchronization: from theory to data analysis. 4:279–321.

Runge, C. (1895). Über die numerische auflösung von differentialgleichungen. *Mathematische Annalen*, 46(2):167–178.

Rutishauser, U., Douglas, R. J., and Slotine, J.-J. (2011). Collective stability of networks of winner-take-all circuits. volume 23, pages 735–773. MIT Press.

Sandamirskaya, Y. (2014). Dynamic neural fields as a step toward cognitive neuromorphic architectures. *Frontiers in Neuroscience*, 7:276.

Schöner, G. (2008). Dynamical systems approaches to cognition. *Cambridge handbook of computational cognitive modeling*, pages 101–126.

Schwalger, T., Deger, M., and Gerstner, W. (2017). Towards a theory of cortical columns: From spiking neurons to interacting neural populations of finite size. *PLoS computational biology*, 13(4):e1005507.

Segev, I., Burke, R. E., and Hines, M. (1989). Compartmental models of complex neurons. *Methods in neuronal modeling*, 63.

Shapiro, M. L. and Ferbinteanu, J. (2006). Relative spike timing in pairs of hippocampal neurons distinguishes the beginning and end of journeys. *Proceedings of the National Academy of Sciences*, 103(11):4287–4292.

Szentágothai, J. (1978). The neuron network of the cerebral cortex: A functional interpretation. the ferrier lecture 1977. *Proc R Soc Lond*, 201:219–248.

Wilson, H. R. and Cowan, J. D. (1972). Excitatory and inhibitory interactions in localized populations of model neurons. *Biophysical journal*, 12(1):1–24.

Wilson, H. R. and Cowan, J. D. (1973). A mathematical theory of the functional dynamics of cortical and thalamic nervous tissue. *Kybernetik*, 13(2):55–80.

Appendix A

Matlab app for phase plane visualization

I built an interactive application in MATLAB to visualize the phase plane and input-output relationship for the 2-unit rate-based model, described in Eqs. 2.102 and 2.103 in the second chapter. In this graphical user interface, a user can explore what parameter exerts what kind of impact on the phase planes and activity. A screenshot is provided in Fig. A.1.

Here, I will provide a brief characterization of the core of the model. The model can be described through two differential equations governing the activity of the excitatory (u_1) and the inhibitory (u_2) unit:

$$\tau \frac{du_1}{dt} = -g_1 u_1 + F(s + \alpha u_1 - \beta_2 u_2 - T_1) \quad \text{(A.1)}$$

$$\tau \frac{du_2}{dt} = -g_2 u_2 + F(\beta_1 u_1 - T_2), \quad \text{(A.2)}$$

where $F(x) = max(0, x)$, g_1 and g_2 represent the conductances, T_1 and T_2 serve as the thresholds for the activation function, s is the external input to the excitatory unit, α is the self-excitation weight, β_1 is the weight from the excitatory to the inhibitory unit, and $-\beta_2$ is the weight from the inhibtory to the excitatory unit. This set of parameters can be changed in the app.

The left panel of Fig. A.1 shows the phase plane for a parameter set given. The nullcline for each unit is calculated by setting the left-hand side of the differential equation to zero:

$$0 = -g_1 u_1 + F(s + \alpha u_1 - \beta_2 u_2 - T_1) \quad \text{(A.3)}$$

$$0 = -g_2 u_2 + F(\beta_1 u_1 - T_2). \quad \text{(A.4)}$$

The arrows in the vector field in the left panel of Fig. A.1 indicate the direction and strength in and with which the system would head if it were placed in

precisely that state. These quantities can be calculated from the right-hand side of Eq. A.1 and A.2 at each grid point.

The activity levels, shown in the right panel of Fig. A.1, are calculated by finding the intersections of the nullclines, i.e., the fixed points. When there are three fixed points, the two outer ones are stable. The one with lower activity is assigned to the forward path; the one with higher activity to the backward path. The dashed red line represents the current input level which corresponds to that on which the phase plane plot in the left panel is based on.

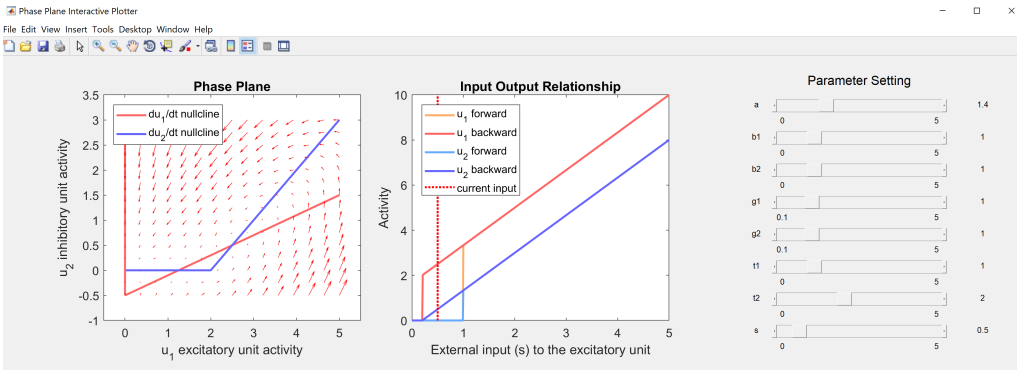


Figure A.1: A Screenshot of interactive phase plane and activity prediction visualization tool. The parameters can be changed by moving the slider, and the graphs are updated dynamically.

Listing A.1: Matlab code for the interactive application.

```
%INTERACTIVE Phase Plane Plotter%
set(0, 'DefaultAxesFontSize', 12);

%Default Parameter Values
a=1.5;
b1=1;
b2=1;
g1=1;
g2=1;
t1=1;
t2=2;
s=0.5;

pnames=["a","b1","b2","g1","g2","t1","t2","s"];
num_p=length(pnames);
p_values=[a;b1;b2;g1;g2;t1;t2;s];

% Slider Range [Min Max]
mm=[0  5; %a
    0  5; %b1
    0  5; %b2
    0.1  5; %g1
    0.1  5; %g2
    0  5; %t1
    0  5; %t2
    0  5];%s

```

```
28 % x1 Plotting Range
29 x1=[0:0.01:5];
30 Num_Arrow=10;
31
32 %%%% UI Part %%%%
33
34 % UI parameters
35 SliderLen=0.17;
36 SliderWid=0.04;
37 SliderLeftL=0.75;
38 SliderInterv=0.1;
39 SliderBtm=0.125;
40 ad10=0.01;
41
42
43 % Plottting Area
44 f = figure('Name','Phase Plane Interactive Plotter','NumberTitle','
       off');
45 f.Units='pixels';
46 SZ=get(0,'ScreenSize');
47 f.Position=[SZ(3)*0.2,SZ(4)*0.1,SZ(3)*0.9,SZ(4)*0.5];
48
49 ax = axes('Parent',f,'position',[0.1  0.18   0.25  0.7]);
50 ax(2) = axes('Parent',f,'position',[0.4  0.18   0.25  0.7]);
51 ints=recalc_replot_activity(p_values,mm,ax(2));
52 recalc_replot(p_values,x1,ax(1),ints);
53 bgcolor = f.Color;
54
55 % Control sliders
56 p_ui = uicontrol('Parent',f,'Units','normalized','Style','text',...
57                  'String','Parameter Setting','FontSize',13,'
                       BackgroundColor',bgcolor);
58 p_ui.Position = [SliderLeftL,SliderBtm+SliderInterv*7.75,SliderLen,
       ad10*5];
59
60 for j=1:num_p
61     p_ui(j+1) = uicontrol('Parent',f,'Units','normalized','Style','
           text','BackgroundColor',bgcolor);
62     p_ui(j+1).Position = [SliderLeftL-ad10*3,SliderBtm+SliderInterv
           *(8-j),ad10*3,SliderWid];
63     p_ui(j+1).String = pnames(j);
64 end
65 for j=1:num_p
66     p_ui(j+num_p+1) = uicontrol('Parent',f,'Units','normalized','
           Style','text','BackgroundColor',bgcolor);
67     p_ui(j+num_p+1).Position = [SliderLeftL+SliderLen-ad10*2,
           SliderBtm-ad10*5+SliderInterv*(8-j),ad10*2,SliderWid];
68     p_ui(j+num_p+1).String =num2str(mm(j,2)) ;
69 end
70 for j=1:num_p
71     p_ui(j+2*num_p+1) = uicontrol('Parent',f,'Units','normalized','
           Style','text','BackgroundColor',bgcolor);
72     p_ui(j+2*num_p+1).Position = [SliderLeftL,SliderBtm-ad10*5+
           SliderInterv*(8-j),ad10*2,SliderWid];
73     p_ui(j+2*num_p+1).String =num2str(mm(j,1)) ;
74 end
75 for j=1:num_p
76     p_ui(j+3*num_p+1) = uicontrol('Parent',f,'Units','normalized','
           Style','text','BackgroundColor',bgcolor);
77     p_ui(j+3*num_p+1).Position = [SliderLeftL+SliderLen+ad10,
           SliderBtm+SliderInterv*(8-j),ad10*5,SliderWid];
78     p_ui(j+3*num_p+1).Tag=strcat('currv_',pnames(j));
79     p_ui(j+3*num_p+1).String =num2str(p_values(j)) ;
```

```
80  end
81  for j=1:num_p
82      p_ui(j+4*num_p+1) = uicontrol('Parent',f,'Units','normalized','
            Style','slider',...
83          'value',p_values(j), 'min',mm(j,1), 'max',mm(j,2),'UserData'
                ,[j p_values(j)]);
84      p_ui(j+4*num_p+1).Position = [SliderLeftL,SliderBtm+SliderInterv
            *(8-j),SliderLen,SliderWid];
85      p_ui(j+4*num_p+1).Tag = strcat('slider_',pnames(j));
86  end
87  for j=1:num_p
88      p_ui(j+4*num_p+1).Callback = @(es,ed) recalc(es,num_p,p_ui,x1,ax
            ,mm);
89  end
90
91  %%%% Functons Implementations %%%%
92
93  % Derivatice calculation function for phase plane
94  function A=calc_derivative(x1,x2,p_values)
95      a=p_values(1);
96      b1=p_values(2);
97      b2=p_values(3);
98      g1=p_values(4);
99      g2=p_values(5);
100     t1=p_values(6);
101     t2=p_values(7);
102     s=p_values(8);
103     A=zeros(1,2);
104     A(1)=-g1*x1+max(0,s+a*x1-b2*x2-t1);
105     A(2)=-g2*x2+max(0,b1*x1-t2);
106 end
107
108 % Phase plane plotting function
109 function recalc_replot(p_values,x1,ax,ints)
110     a=p_values(1);
111     b1=p_values(2);
112     b2=p_values(3);
113     g1=p_values(4);
114     g2=p_values(5);
115     t1=p_values(6);
116     t2=p_values(7);
117     s=p_values(8);
118
119     if ints>4
120         x1=linspace(0,ints*1.3,500);
121     end
122
123     x2_2=(s+(a-g1).*x1-t1)/b2;
124     plot(ax,x1,x2_2,'color',[1,100/255,100/255],'Linewidth',2);
125     hold(ax,'on');
126     idx=(b1.*x1-t2)>0;
127     x2_1=zeros(length(x1),1);
128     x2_1(idx)=(b1.*x1(idx)-t2)/g2;
129     plot(ax,x1,x2_1,'color',[100/255,100/255,1],'Linewidth',2);
130
131     min_y=min(min(x2_1),min(x2_2));
132     max_y=max(max(x2_1),max(x2_2));
133     y1 = linspace(min(x1),max(x1),15);
134     y2 = linspace(min_y,max_y,15);
135
136     plot(ax,[0 0],[x2_2(1) max_y],'color',[1,100/255,100/255],'
            Linewidth',2);
137     [x,y] = meshgrid(y1,y2);
```

```
138     u = zeros(size(x));
139     v = zeros(size(x));
140
141     for i = 1:numel(x)
142         D = calc_derivative(x(i),y(i),p_values);
143         u(i) = D(1);
144         v(i) = D(2);
145     end
146     quiver(ax,x,y,u,v,'r');
147     hold(ax,'off');
148     xlim(ax,[-0.5 max(x1)+0.5])
149     ylim(ax,[min_y-0.5,max_y+0.5])
150     title(ax,"Phase Plane");
151     xlabel(ax,'u_1 excitatory unit activity')
152     ylabel(ax,'u_2 inhibitory unit activity')
153     legend(ax,'du_1/dt nullcline','du_2/dt nullcline','Location','
            northwest')
154 end
155
156 % Parameter value update function
157 function p_values=update_params(num_p,p_ui)
158     p_values=zeros(num_p,1);
159     for j=1:num_p
160         p_values(j)=p_ui(j+4*num_p+1).UserData(2);
161     end
162 end
163
164 % Activity calculation function
165 function ints=recalc_replot_activity(p_values,mm,ax)
166     a=p_values(1);
167     b1=p_values(2);
168     b2=p_values(3);
169     g1=p_values(4);
170     g2=p_values(5);
171     t1=p_values(6);
172     t2=p_values(7);
173     s=p_values(8);
174
175     xs=[mm(8,1):0.01:mm(8,2)];
176     if (a-g1)>0
177         if (b1*b2+(g1-a)*g2)==0
178             u1_ba=-1;
179         else
180             u1_ba=((xs-t1).*g2+b2*t2)/(b1*b2+(g1-a)*g2);
181         end
182         idx=b1.*u1_ba-t2<0;
183         u1_ba(idx)=0;
184         idx1=(xs-t1)/(g1-a)<0;
185         u1_ba(idx&idx1)=-1;
186     elseif (a-g1)==0
187         u1_ba=((xs-t1).*g2+b2*t2)/(b1*b2+(g1-a)*g2);
188         idx=b1.*u1_ba-t2<0;
189         u1_ba(idx)=0;
190     else
191         u1_ba=((xs-t1).*g2+b2*t2)/(b1*b2+(g1-a)*g2);
192         idx=b1.*u1_ba-t2<0;
193         u1_ba(idx)=(xs(idx)-t1)/(g1-a);
194         idx=u1_ba<0;
195         u1_ba(idx)=0;
196     end
197
198     u2_ba=max(0,b1.*u1_ba-t2);
199     u2_ba(u1_ba==-1)=-1;
```

```
200
201         idx=(xs-t1)./b2<0;
202         u1_fo=u1_ba;
203         u1_fo(idx)=0;
204         u2_fo=max(0,b1.*u1_fo-t2);
205         u2_fo(u1_fo==-1)=-1;
206
207         if (b1*b2+(g1-a)*g2)>0
208             ints=((s-t1).*g2+b2*t2)/(b1*b2+(g1-a)*g2);
209         else
210             ints=-1;
211         end
212
213
214         plot(ax,xs,u1_fo,'Linewidth',2,'color',[1,171/255,100/255]);
215         hold(ax,'on');
216         plot(ax,xs,u1_ba,'Linewidth',2,'color',[1,100/255,100/255]);
217         plot(ax,xs,u2_fo,'Linewidth',2,'color',[100/255,171/255,1]);
218         plot(ax,xs,u2_ba,'Linewidth',2,'color',[100/255,100/255,1]);
219         y_max=max(max(u1_ba),max(u2_ba));
220         if y_max==0
221             y_max=mm(8,2);
222         end
223
224         if(sum(u1_ba==-1)>0)
225             plot(ax,[s s],[-1 y_max],':r','Linewidth',2);
226             ylim([-1 y_max]);
227         else
228             plot(ax,[s s],[0 y_max],':r','Linewidth',2);
229             ylim([0 y_max]);
230         end
231         hold(ax,'off');
232         legend(ax,'u_1 forward','u_1 backward','u_2 forward','u_2
                backward',...
233             'current input','Location','northwest')
234         xlabel("External input (s) to the excitatory unit");
235         ylabel("Activity");
236         title("Input Output Relationship");
237
238 end
239
240 % Callback function
241 function recalc(es,num_p,p_ui,x1,ax,mm)
242         es.UserData(2)=es.Value;
243         p_ui(es.UserData(1)+3*num_p+1).String=num2str(es.Value);
244         p_values=update_params(num_p,p_ui);
245         ints=recalc_replot_activity(p_values,mm,ax(2));
246         recalc_replot(p_values,x1,ax(1),ints);
247 end
```

Appendix B

Numerical fixed point approximation program

In order to numerically approximate fixed points for interacting spiking neuron groups, I wrote a program that follows a gradient of the vector field in the phase plane of the system and detects fixed points when it does not move anymore. This program can be used for arbitrary number of units with any types of connections by handing the number of units and the corresponding weight matrix as arguments. As part of this program, I applied the (classical) Runge-Kutta method (Kutta, 1901, Runge, 1895).

To allow for fast computation, I wrote the program in C and compiled it to a dynamic link library (DLL) file (in the form of *.dll*, executable using the Windows operating system) in order to call the function from Python using the *ctypes* library. Note that setting types and proper transformation of variables is required. To make the program easier to use it, I developed a wrapping Python function that calls the C function using the dll (List.B.3).

Here, I wish to describe the parameters specific to this numerical approximation function and their values in the usage in this thesis:

(1) The update coefficient: delta (δ). Let $u_i[t]$ be the value for unit i at the time step t. In the next time step, the update can be described as follows:

$$u_i[t+1] = u_i[t] + \delta(\tau \frac{du_i[t]}{dt}), \tag{B.1}$$

where δ defines the size of the update step. When this value is large, the convergence to the fixed point is fast. When the size of the update step is too large, however, precision problems can emerge. In the usage of this program in this thesis, I set: $\delta = 5.0 \times 10^{-4}$

(2) The convergence judgement border: epsilon (ε) Let $\mathbf{g}[\mathbf{t}]$ be the vector of gradients for all units in the network at time step t. The numerical approximation is judged to be converged when:

$$||\mathbf{g}[\mathbf{t}]|| < \epsilon. \tag{B.2}$$

In this thesis, $\epsilon = 1.0 \times 10^{-6}$

(3) The maximum number of steps: max_size. Sometimes it can take long for the system to converge or not converge at all when the parameter set is not in the stable area. In those cases, it is important to stop the update at some point by setting the maximum number of update steps. In this thesis, max_size=10^5.

Listing B.1: fixedpointRK.c: the main file

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "fixedpoint.h"
#define PI 3.14159265359


// F-F function (without ceiling function) calculation
double ffcurve(paramset param, double f, double w){
    double freqout, a,p;

    p=f/(f+1/param.tau);

    if(f>0&&w>0){
        a=1-param.v_th*(1-p)/w;
    }else{
        a=0;
    }
    if(a>0){
        freqout=1/(param.t_ref+(log(a)/log(p))/f);
    }else{
        freqout=0;
    }
    return freqout;
}

// Slope calculation with the Runge-Kutta method
int calc_defferential_RK(double grad[], paramset param, double w_mat
    [],double x_prev[], double f_ext[], int group_size[]){

    int i,j;
    double diff, w, f;
    int N = param.num_groups;
    double p = param.con_p;
    double(*k)[4];
    double(*temp)[3];
    double *gp;

    k = (double(*)[4])malloc(sizeof(double)*N*4);
    temp = (double(*)[3])malloc(sizeof(double)*N*3);
    gp=malloc(sizeof(double)*N);

    for(j=0;j<N;j++){
        gp[j]=(double)(group_size[j])*p;
    }

    for (j=0;j<N;j++){
        w=0;f=0;
        for (i=0;i<N;i++){
```

```
49 |                 w+=w_mat[j*(N+1)+i]*gp[i]*x_prev[i];
50 |                 f+=gp[i]*x_prev[i];
51 |             }
52 |             w+=w_mat[j*(N+1)+N]*f_ext[j];
53 |             f+=f_ext[j];
54 |
55 |             if(f>0){
56 |                 w/=f;
57 |             }
58 |             k[j][0]=-x_prev[j]+ffcurve(param, f, w);
59 |             temp[j][0]=x_prev[j]+0.5*param.delta*k[j][0];
60 |         }
61 |         for (j=0;j<N;j++){
62 |             w=0;f=0;
63 |             for (i=0;i<N;i++){
64 |                 w+=w_mat[j*(N+1)+i]*gp[i]*temp[i][0];
65 |                 f+=gp[i]*temp[i][0];
66 |             }
67 |             w+=w_mat[j*(N+1)+N]*f_ext[j];
68 |             f+=f_ext[j];
69 |
70 |             if(f>0){
71 |                 w/=f;
72 |             }
73 |             k[j][1]=-temp[j][0]+ffcurve(param, f, w);
74 |             temp[j][1]=x_prev[j]+0.5*param.delta*k[j][1];
75 |         }
76 |         for (j=0;j<N;j++){
77 |             w=0;f=0;
78 |             for (i=0;i<N;i++){
79 |                 w+=w_mat[j*(N+1)+i]*gp[i]*temp[i][1];
80 |                 f+=gp[i]*temp[i][1];
81 |             }
82 |             w+=w_mat[j*(N+1)+N]*f_ext[j];
83 |             f+=f_ext[j];
84 |
85 |             if(f>0){
86 |                 w/=f;
87 |             }
88 |             k[j][2]=-temp[j][1]+ffcurve(param, f, w);
89 |             temp[j][2]=x_prev[j]+param.delta*k[j][2];
90 |         }
91 |         for (j=0;j<N;j++){
92 |             w=0;f=0;
93 |             for (i=0;i<N;i++){
94 |                 w+=w_mat[j*(N+1)+i]*gp[i]*temp[i][2];
95 |                 f+=gp[i]*temp[i][2];
96 |             }
97 |             w+=w_mat[j*(N+1)+N]*f_ext[j];
98 |             f+=f_ext[j];
99 |
100|             if(f>0){
101|                 w/=f;
102|             }
103|             k[j][3]=-temp[j][2]+ffcurve(param, f, w);
104|         }
105|         for (j=0;j<N;j++){
106|             grad[j]=(k[j][0]+2*k[j][1]+2*k[j][2]+k[j][3])/6;
107|         }
108|
109|         free(k);
110|         free(gp);
111|         free(temp);
```

```
112     return 0;
113 }
114
115
116 // Fixed point search by following the gradient
117 int fixedpoint_search(double cord[], double x[], paramset param,
        double def_cor[], double w_mat[],double f_ext[],int group_size
        []){
118
119     int t,i;
120     double *x_prev, *grad;
121     double difference;
122     int N = param.num_groups;
123
124
125     x_prev = malloc(sizeof(double) * N);
126     grad = malloc(sizeof(double)*N);
127
128     for (i=0;i<N;i++){
129         x[i]=def_cor[i];
130         x_prev[i]=x[i];
131     }
132
133     for (t=1;t<param.max_size;t++){
134         difference=0;
135         calc_defferential_RK(grad,param,w_mat,x_prev,f_ext,
                group_size);
136         for (i=0;i<N;i++){
137             x[t*N+i]=x[(t-1)*N+i]+param.delta*grad[i];
138             difference+=(x[t*N+i]-x_prev[i])*(x[t*N+i]-x_prev[i]);
139             if (x[t*N+i]<0){
140                 x[t*N+i]=0;
141             }
142             x_prev[i]=x[t*N+i];
143         }
144         difference=sqrt(difference);
145         if (difference<param.epsilon){
146             break;
147         }
148     }
149     for (i=0;i<N;i++){
150         cord[i] = x_prev[i];
151     }
152
153     free(x_prev);
154     free(grad);
155
156     return t;
157 }
158
159 // Main function in the dll form
160 __declspec(dllexport) int activity_prediction(int flag[], double
        cord_all[], double x[], paramset param, double def_cor[],
        double w_mat[],double f_ext[],int group_size[]){
161
162     int freq,f;
163     int N=param.num_groups;
164
165     freq=0;
166     flag[freq]=fixedpoint_search(&cord_all[N*freq],x,param,&def_cor
            [0],w_mat,&f_ext[N*freq],group_size);
167     for (freq=1;freq<param.f_ext_res;freq++){
168         flag[freq]=fixedpoint_search(&cord_all[N*freq],x,param,&
```

```
                cord_all[N*freq-N],w_mat,&f_ext[N*freq],group_size);
169     }
170     for (freq=0;freq<param.f_ext_res;freq++){
171         flag[param.f_ext_res+freq]=fixedpoint_search(&cord_all[N*(
                param.f_ext_res+freq)],x,param,&cord_all[N*(param.
                f_ext_res+freq)-N],w_mat,&f_ext[N*(param.f_ext_res-freq)
                -N],group_size);
172     }
173
174     return 0;
175 }
```

Listing B.2: fixedpoint.h: the headder file

```
1  #ifndef __FIXEDPOINT_H__
2  #define __FIXEDPOINT_H__
3
4  #include <math.h>
5
6  typedef struct parameter_set{
7      int num_groups;
8      double con_p;
9
10     double tau;
11     double v_th;
12     double t_ref;
13
14     double epsilon;
15     double delta;
16
17     int max_size;
18
19     int f_ext_res;
20 } paramset;
21
22
23 __declspec(dllexport) int activity_prediction(int flag[], double
       cord_all[], double x[], paramset param, double def_cor[], double
       w_mat[],double f_ext[],int group_size[]);
24
25
26 #endif
```

Listing B.3: fixedpoint.py: the wrapper python function

```
1  from ctypes import *
2  from brian2 import second
3  import numpy as np
4
5
6  # Struct to be passed to the C function.
7  class Paramset(Structure):
8      _fields_ = [
9          ("num_groups",c_int),
10         ("con_p",c_double),
11         ("tau",c_double),
12         ("v_th",c_double),
13         ("t_ref",c_double),
14         ("epsilon",c_double),
15         ("delta",c_double),
16         ("max_size",c_int),
```

```
17          ("f_ext_res",c_int)]
18
19
20  # Python interface function for calling the C function.
21  def calc_numerical(params, stim,  w_mat, lib):
22
23      param = Paramset()
24      param.num_groups = params['num_exc_pop'] + params['num_inh_pop']
25      param.con_p = params['conn_p']
26      param.tau = params['tau'] / second
27      param.v_th = params['thr']
28      param.t_ref = params['refp'] / second
29
30      param.epsilon = params['epsilon']
31      param.delta = params['delta']
32      param.max_size = params['max_size']
33      param.f_ext_res = params['f_ext_res']
34
35      N = param.num_groups
36
37      f_ext_min = np.append(np.tile(stim.stim_min, params['num_exc_pop
            ']), np.tile(0, params['num_inh_pop']))
38      f_ext_max = np.append(np.tile(stim.stim_max, params['num_exc_pop
            ']), np.tile(0, params['num_inh_pop']))
39
40      def_cor = np.tile(0, N)
41
42      group_size = np.tile(params['num_per_pop'], N)
43
44      f_ext = [0 for i in range(N * param.f_ext_res)]
45
46      for i in range(param.f_ext_res):
47          for j in range(N):
48              f_ext[N * i + j] = f_ext_min[j] + (f_ext_max[j] -
                    f_ext_min[j]) * i / param.f_ext_res
49
50      IntFlag = c_int * (2 * param.f_ext_res)
51      DoubleCord = c_double * (N * 2 * param.f_ext_res)
52      DoubleX = c_double * (N * param.max_size)
53
54      DoubleDefcor = c_double * N
55      DoubleWmat = c_double * ((N + 1) * N)
56      DoubleFext = c_double * (N * param.f_ext_res)
57      IntGsize = c_int * N
58
59      flag = [0 for i in range(2 * param.f_ext_res)]
60      cord_all = [0 for i in range(N * 2 * param.f_ext_res)]
61      x = [0 for i in range(N * param.max_size)]
62
63      flag_c = IntFlag(*flag)
64      cord_all_c = DoubleCord(*cord_all)
65      x_c = DoubleX(*x)
66      def_cor_c = DoubleDefcor(*def_cor)
67      w_mat_c = DoubleWmat(*w_mat)
68      f_ext_c = DoubleFext(*f_ext)
69      group_size_c = IntGsize(*group_size)
70
71      lib.activity_prediction.argtypes = [c_void_p, c_void_p, c_void_p
            , Structure, c_void_p, c_void_p, c_void_p, c_void_p]
72      lib.activity_prediction.restype = c_int
73
74      lib.activity_prediction(pointer(flag_c), byref(cord_all_c),
            byref(x_c), param, byref(def_cor_c), byref(w_mat_c),
```

```
75                                      byref(f_ext_c), byref(group_size_c))
76      f_ext_np = np.array(f_ext).reshape((param.f_ext_res, param.
            num_groups))
77      flag_np = np.array(flag_c).reshape((param.f_ext_res, 2))
78      cord_all_np = np.array(cord_all_c).reshape((2, param.f_ext_res,
            param.num_groups))
79      cord_all_np[1, :, :] = np.flip(cord_all_np[1, :, :], axis=0)
80
81      return f_ext_np, cord_all_np, flag_np
```