Biological and Functional Models of Learning in Networks of Spiking Neurons

by

Lars Holger Büsing

ISSERTATION

submitted for the degree of

Doctor Rerum Naturalium

Institute for Theoretical Computer Science
Graz University of Technology

Thesis Advisor: Univ. Prof. DI Dr. Wolfgang Maass

defended on August 24, 2010

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbständig verfasst, andere als die angegebenen Quellen / Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Graz, 1st August 2010 .....

Lars Büsing

Abstract

Neural circuits generally process information in a massively parallel way and exhibit a communication between the constituent units based on spikes, i.e. binary events, therefore differing fundamentally from many artificial information processing and learning systems. In such neural circuits, synaptic plasticity is widely considered to be the main biophysical correlate of learning. This thesis investigates synaptic plasticity and learning in neural networks with the help of data-driven, i.e. "bottom-up", and theory-driven, i.e. "top-down", models and focuses in particular on the implications of their distributed architecture and spike-based communication for learning.

In Chapter 2, a novel model of experimental data on synaptic plasticity, unifying multiple previous models, is presented. The proposed model is able to reproduce the experimentally observed effect of spike timing-dependent plasticity as well as plasticity effects parametrized by postsynaptic firing rate or depolarization.

Chapters 3 and 4 propose learning rules that enable spiking neurons to perform clustering of input data taking into account side-information. These rules, which implement the Information Bottleneck method in spiking networks, are designed to operate in distributed architectures exclusively using biologically plausible communication mechanisms.

In Chapter 5 and 6, the capabilities of recurrent neural networks as multi-purpose preprocessors for diverse learning problems are studied. In this context, essential differences between spiking and non-spiking network models are revealed especially with respect to the influence of the network connectivity statistics on the preprocessing capabilities.

Zusammenfassung

Neuronale Schaltkreise verarbeiten Informationen zumeist auf eine massiv parallele Weise und weisen eine spikebasierte Kommunikation zwischen den einzelnen Teilstrukturen auf, worin sie sich sich fundamental von vielen künstlichen Datenverarbeitungs- und Lernsystemen unterscheiden. Synaptische Plastizität in dieser Art neuronaler Netze wird als wichtigstes Korrelat von Lernprozessen angesehen. In dieser Dissertation wird synaptische Plastizität und Lernen in spikenden, neuronalen Netzwerken mit Hilfe von datengestützten, sog. "botton-up", sowie theoretischen, sog. "top-down", Modellen untersucht mit besonderem Augenmerk auf die Auswirkungen ihrer parallelen Architektur und spikebasierten Kommunikation.

In Kapitel 2 wird ein neues Plastizitätsmodell, welches direkt experimentelle Daten beschreibt, vorgestellt. Dieses Modell kann erfolgreich sowohl Plastizitätseffekte, die durch Spikezeiten als auch solche, die durch die postsynaptische Feuerrate oder Depolarisation parametrisiert werden, reproduzieren.

In Kapitel 3 und 4 werden Lernregeln eingeführt, die Clustering unter Berücksichtigung von Nebeninformationen in spikenden neuronalen Netzen implementieren. Diese Lernregeln, welche aus dem Information Bottleneck Formalismus abgeleitet sind, sind so entworfen, dass sie in verteilten Systemen mithilfe ausschließlich biologisch realistischer Kommunikationskanäle funktionieren.

In Kapitel 5 und 6 wird die Leistungsfähigkeit rekurrenter neuronaler Netze als Mehrzweckpräprozessoren für verschiedene Lernprobleme untersucht. In diesem Zusammenhang werden tiefgreifende Unterschiede zwischen spikenden und nichtspikenden Netzwerkmodellen aufgezeigt, im Besonderen bezüglich des Einflusses der Netzwerkverbindungsstatistik auf die Leistungsfähigkeit als Präprozessor.

Acknowledgments

I want to thank my advisor Wolfgang Maass, for inspiring my research and for his continuous support.
I am also very grateful to Benjamin Schrauwen, Wulfram Gerstner and Eleni Vasilaki for the fruitful collaborations and their guidance.
Furthermore I would like to show my gratitude to my friends and former colleagues that I met at the EPFL and Sheffield University.
My thanks also go to my colleagues at the TU Graz for making the stay in beautiful Austria unforgettable.
I am grateful for the financial support provided by the research programs of the European Union.
Last but not least, I would like to thank my parents and my sister as well as my friends and you, Emma, for massively supporting me.

Contents

Chapter 1

Introduction

Seen through the eyes of a scientist, the human brain is probably the most remarkable agglomeration of matter known. It is of extraordinary complexity and flexibility, which becomes manifest in the brain's unsurpassed abilities to react and adapt to sensory input data. These reactions and adaptations are highly nontrivial processes that take place on many different time scales. On the one end of the spectrum, i.e. on time scales of less than a second, e.g. sensory processing is able to adapt to fluctuations of the environmental background conditions and motor responses can be initiated to react to sensory cues. On the other end of the spectrum, i.e. on time scales extending over years or decades, e.g. memories can be maintained and precisely recalled at suitable points in time, whereas the storing process itself only requires the brain to be exposed to the sensory stimuli to be memorized for as few as some seconds. Cerebral properties like the ones mentioned above, give rise to the astonishing human capabilities to carry out complex computations (e.g. to anticipate trajectories of moving objects or to weight "pros" and "cons" of an intended action) as well as to solve complex learning tasks (e.g. to learn categories of objects as well as highly abstract concepts). Surprisingly many of theses information processing tasks, being highly demanding for artificial systems, are carried out by humans online without conscious effort. Developing suitable mathematical concepts and understanding the biological mechanisms that underlying the impressive cognitive capabilities of humans is of greatest scholarly interest as well as of immense technological relevance.

The agenda of Computational Neuroscience, i.e. the scientific effort to quantitatively understand neural information processing such as computations, adaptation and learning, is largely based on the fundamental hypothesis that neural structures and processes serve a *purpose*, i.e. that questions such as "What is the functionality of this neural structure?" or "Why do these processes interact in this specific, experimentally observed way?" are actually scientifically sensible. To see the relevance and impact of the aspect of purpose (shared among all Life Sciences), one has only to compare Computational Neuroscience with Physics. In the latter discipline answers to questions containing "Why" never refer to a purpose but always point to "lower", more fundamental levels of description, of which the lowest does not offer a sensible answer to "Why?"-questions apart from pointing to experimental measurements. The notion of purpose in Life Sciences itself can of course be grounded in Darwin's Evolutionary Theory. The principles of variation and selection imply that physiological structures and processes, being costly to grow and maintain, have to increase the fitness of the agent in question. Therefore, many neural structures are considered to be extrema, i.e. optimal solutions, in the "fitness landscape", the latter being shaped by the task these neural structures are hypothesized to solve. A research program in Computational Neuroscience that makes

use of the principle of evolutionary purpose, such as the program underlying this thesis, is partially similar to engineering work (which can also be characterized as finding near-optimal solutions to certain problems) in the following sense. The researcher takes a task or problem that a biological agent faces and tries to solve this problem in an optimal way (the engineering aspect). He then compares his solution, *the model*, with experimental data on the "evolutionary solution" existing in the agent. In case no major discrepancies are noticed, the researcher has found a valid hypothesis about the functionality (the purpose) and the mode of operation of the studied neural system. Then, further experiments may strengthen the belief in or falsify this hypothesis. This method based on the "optimality-hypothesis" is very powerful for analyzing neural systems and will also be used in the theoretical part of this thesis presented in Chapters 3-6.

However, from the mathematical discipline of optimization as well as from engineering, it is well known that it is crucial to also take into account *constraints* that possible solutions of optimization problems have to satisfy. Indeed, all engineering problems trivially only make sense if they are posed with sensible constraints. These constraints can be of different forms, they can be either formalized as hard constraints, i.e. equality or inequality constraints, or soft constraints, which are associated to costs. Analogously to engineering, "evolutionary optimization" is also subject to hard and soft constraints of course. As a trivial example one might imagine that a superior neural system could be designed based on Tantalum (a very scarce metal), however poor availability would make an evolutionary step in this direction impossible. Thus, a profound understanding of biological systems requires taking into account the environmental constraints these systems are subject to. Unfortunately, for many evolutionary situations it is impossible to obtain exact knowledge of the constraints merely due to their amount and complexity, reflecting the full complexity of the environment in which evolutionary processes often take place. This problem is a serious obstacle for modelling and understanding biological systems with the help of the "optimality-hypothesis". This especially holds for neural systems, such as the ones considered in this thesis, as their evolutionary purposes and environmental conditions are particularly complex.

An approach that is often used to overcome the problem of largely unknown evolutionary constraints is the following. When modelling a certain neural systems in order to determine their purpose and functionality, instead of explicitly modelling the unknown constraints, one often directly constraints the set of considered mathematical models to those that feature characteristic properties of this neural system, such as its anatomical architecture. In Computational Neuroscience e.g. , connectionist models are often considered, as these models reflect the fact that biological neural system are distributed information processing systems consisting of units which communicate over low-capacity channels, without knowing the evolutionary reasons that actually lead to this massively parallel architecture. This general procedure is motivated by the consideration that all models in the constraint model set fulfill the evolutionary constraints which are largely unknown. Two important constraints of neural systems, which were used to constrain the set of models considered in thesis and which are hence in the focus of the investigation presented here, are described in the following two paragraphs.

This first constraint is the following. Earliest anatomical and electrophysiological

findings regarding nervous systems reveal that they process information in an highly distributed way. In contrast to common personal computer architectures, having a central processing unit, the visual system of vertebrates e.g. works in an essentially non-sequential, parallel manner. The main problem that comes with mathematical models of distributed learning systems is the fact that information, that is needed for updating model parameters during learning, must be available at the locations of the parameters. This constraint is often referred to as *locality of information*. Consider e.g. the problem of matrix inversion. Stating an abstract algorithm (e.g. the recursive scheme known as "Cramer's rule") for this problem is rather simple compared to actually designing circuits that perform matrix inversion purely locally with respect to their spatial (mostly "two-plus-epsilon" dimensional) embedding (Singh, Prasad, & Balsara, 2007). As can be seen from this example, possible hypotheses for computations and learning in local, distributed systems seriously differ from unconstrained hypotheses and require additional modelling effort.

The second constraint on neural information processing models applied in this thesis, is the fact that neurons, often considered the "atomic" units of information processing, communicate via *binary, stereotyped electric events*, so-called action potentials or spikes[1]. Although been known long before the work of Hodgkin and Huxley (Hodgkin & Huxley, 1952), the binary nature of biological neural networks was often neglected in early models in favor of continuous models due to their advantageous mathematical properties. Spiking models of neural networks (additionally having an inherent temporal aspect), are indeed profoundly different from continuous (static) models and therefore demand different (or at least highly adapted) mathematical hypothesis for computations and learning. Error-Backpropagation (Rumelhart, Hinton, & Williams, 1986) in its initial form is a typical example of an implausible learning hypothesis for spiking networks, as it requires $\mathcal{C}^1$ (continuously differentiable) activation functions, and could only be rendered a plausible hypothesis with serious alternations.

The considerations above illustrate that in order to profoundly understand the purpose and functionality of neural systems, it is essential to sensibly constrain the set of models to those reflecting the most crucial anatomical and electrophysiological characteristics of biological neural circuits. The main theme of this thesis is to investigate the phenomenon of learning in neural networks using "bottom-up" models of experimental data on synaptic plasticity (widely believed to be the main biophysical correlate of learning), and theoretical optimality-models, so-called "top-down" models. The thesis focuses in particular on the consequences of the constraints imposed by the distributed architecture and the communication via electrical "all-or-nothing" events of biological neural circuits.

Organization of the Thesis

In Chapter 2, a novel model of experimental data on synaptic plasticity is presented, which unifies multiple previous models. Synaptic plasticity is widely considered to be the

[1]There are non-spiking channels of communication between neurons, such as gap junctions and neuromodulators. These are however not subject of this thesis.

most important biophysical mechanism which implements learning. Early experimental studies, in accordance with the theoretical view at that time that neurons transmit a noisy rate (i.e. a continuous variable), investigated phenomena of synaptic plasticity using experimental protocols that did not directly explore the discontinuous (action potential) nature of neural information processing and its impact on learning. They were rather based on eliciting synaptic plasticity as functions of continuous variables such as postsynaptic firing rate or postsynaptic intracellular variables, e.g. depolarization. More recent experimental studies of synaptic plasticity heavily focused on the significance of the action potential-based transmission of information for learning and culminated in the finding of *spike timing-dependent plasticity*. This plasticity effect fundamentally reflects the two crucial constraints of biological neural networks mentioned above, namely the binary and distributed nature of neural information processing. However, a simple mathematical model of plasticity that is able to reproduce results from both classes of stimulation protocols, spike-based on the one hand and rate / depolarization-based protocols on the other, was missing until now. A precise and concise model is presented that is able to unify the description of both plasticity aspects, aimed at further fostering the understanding of the mechanism underlying learning in binary, distributed neural networks.

Chapter 3 presents a theoretically motivated "top-down" model of synaptic plasticity (that is especially suited for distributed, spiking circuits) implementing clustering with side-information in neural networks. Unsupervised learning, such as clustering, has often been hypothesized as a natural learning goal for neural circuits. Multiple algorithms for continuous and spiking networks have been derived that enable neurons to learn lower dimensional representations of input data. Based on previous work, a simple learning rule is presented and analyzed that implements clustering with side-information in neurons by learning the neural weights according to the Information Bottleneck (IB) method. The IB framework, which is based on maximization of mutual information, is especially well suited as a model for learning in probabilistic spiking networks (in contrast to continuous network models) as the IB objective function in this case is a "well-behaved", bounded function of the networks parameters.

In Chapter 4 the IB approach for spiking circuits is extended to allow neurons to perform clustering in the case of more complex input distributions. To demonstrate the flexibility of the extended IB learning rule, it is applied to enable neurons to learn a so-called predictive coding, i.e. a coding where neurons learn to predict their own future output. Using such a predictive coding scheme, neurons learn to extract components from their input that exhibit a high level of temporal stability, a learning objective that is thought to be useful to extract relevant and meaningful features of sensory input such as the presence and categories of objects. Furthermore, this chapter briefly reviews and discusses experimentally observed plasticity mechanisms that would in principle allow neural circuits to learn IB optimal clustering in a completely *local* way, rendering the IB framework a plausible hypothesis for learning in spiking neural circuits.

In many machine learning applications, the performance of a learning algorithm on a certain task crucially depends on a suitable preprocessing of the input data. Supposedly this is also the case for learning problems that neural systems face. In Chapter 5 of this

thesis, the capabilities of neural networks as multi-purpose preprocessors for various learning tasks are investigated. This study focuses in particular on the differences between preprocessor networks that consist of binary (spiking) neuron models compared to analog neuron models. This approach sheds light on the consequences of the binary and distributed nature of neural circuits in the context of preprocessors. Furthermore, the influence of the network dynamics, which are strongly shaped by different connectivity statistics, on the preprocessing capabilities is studied. It is shown that an intimate relationship between characteristics of network dynamics, such as the Lyapunov exponent, and the preprocessing capabilities exists.

In Chapter 6 the investigation of the differences between analog and spiking network models as preprocessors is extended. It is shown that important characteristic quantities of the network dynamics of binary as well as of analog networks can be faithfully approximated analytically. These approximations can then be utilized to identify suitable regimes of network parameters for preprocessing, revealing stark differences between binary and analog network models. Additionally, a novel upper bound for the network memory capacity is derived, further illustrating the close connection between network dynamics and preprocessing capabilities of recurrent neural networks.

Chapter 2

Connectivity Reflects Coding: A Model of Voltage-Based Spike Timing-Dependent Plasticity

Contents

Electrophysiological connectivity patterns in cortex often show a few strong connections in a sea of weak connections. In some brain areas a large fraction of strong connections are bidirectional, in others they are mainly unidirectional. In order to explain these connectivity patterns, we use a model of spike timing-dependent plasticity where synaptic changes depend on presynaptic spike arrival and the postsynaptic membrane potential, filtered with two different time constants. The model describes several nonlinear effects in STDP experiments, as well as the voltage dependence of plasticity under voltage clamp and classical paradigms of LTP/LTD induction. We show that in a simulated recurrent network of spiking neurons our plasticity rule leads not only to development of localized receptive fields, but also to connectivity patterns that reflect the neural code: for temporal coding paradigms with spatio-temporal input correlations, strong connections are predominantly unidirectional, whereas they are bidirectional under rate coded input with spatial correlations only. Thus variable connectivity patterns in the brain could reflect different coding principles across brain areas; moreover our simulations suggest that rewiring the network can be surprisingly fast.

2.1 Introduction

Experience-dependent changes in receptive fields (Buonomano & Merzenich, 1998; Fregnac & Shulz, 1999; Froemke, Merzenich, & Schreiner, 2007) or in learned behavior (Recanzone, Schreiner, & Merzenich, 1993) may occur through changes in synaptic strength. Thus, electrophysiological measurements of functional connectivity patterns in slices of neural tissue (Song, Sjöström, Reigl, Nelson, & Chklovskii, 2005; Lefort, Tomm, Sarria, & Petersen, 2009) or anatomical connectivity measures (Denk & Horstmann,

2004) can only present a snapshot of the momentary connectivity – which may change with the next set of stimuli. Indeed, modern imaging methods show that spine motility can lead to a rapid rewiring of the connectivity pattern (Yuste & Bonhoeffer, 2004; Trachtenberg et al., 2002) by formation of new synapses or by strengthening or weakening of existing synapses. The question then arises whether the connectivity patterns and changes that are found in experiments can be connected to basic rules of synaptic plasticity, in particular to modern or traditional forms of Hebbian plasticity (Hebb, 1949) such as Long-Term Potentiation and Depression (Malenka & Bear, 2004).

Long-term potentiation LTP and depression LTD of synapses depends on the exact timing of pre- and postsynaptic action potentials (Markram, Lübke, Frotscher, & Sakmann, 1997; Bi & Poo, 2001), but also on postsynaptic voltage (Artola, Bröcher, & Singer, 1990; Ngezahayo, Schachner, & Artola, 2000), and presynaptic stimulation frequency (Dudek & Bear, 1993). spike timing-dependent plasticity (STDP) has attracted particular interest in recent years, since temporal coding schemes where information is contained in the exact timing of spikes rather than mean frequency could be learned by a neural system using STDP (Gerstner, Kempter, van Hemmen, & Wagner, 1996; Roberts & Bell, 2000; Legenstein, Naeger, & Maass, 2005; Guyonneau, VanRullen, & Thorpe, 2005; Gerstner & Kistler, 2002). However, the question whether STDP is more fundamental than frequency dependent plasticity or voltage dependent plasticity rules has not been resolved, despite an intense debate (Lisman & Spruston, 2005). Moreover it is unclear how the interplay of coding and plasticity yield the functional connectivity patterns seen in experiments. In particular, the presence or absence of bidirectional connectivity between cortical pyramidal neurons seems to be contradictory across experimental preparations in visual (Song et al., 2005) or somatosensory cortex (Lefort et al., 2009).

Recent experiments have shown that STDP is strongly influenced by postsynaptic voltage before action potential firing (Sjöström, Turrigiano, & Nelson, 2001), but could not answer the question whether spike timing dependence is a direct consequence of voltage dependence, or the manifestation of an independent process. In addition, STDP depends on stimulation frequency (Sjöström et al., 2001) suggesting an interaction between timing and frequency dependent processes – or this interaction could be the manifestation of a single process in different experimental paradigms. We show that a simple Hebbian plasticity rule that pairs presynaptic spike arrival with the postsynaptic membrane potential is sufficient to explain STDP and the dependence of plasticity upon presynaptic stimulation frequency. Moreover, the intricate interplay of voltage and spike timing dependence seen in experiments (Sjöström et al., 2001) as well as the frequency dependence of STDP can be explained in our model from one single principle. In contrast to earlier attempts towards a unified description of synaptic plasticity rule that focused on detailed biophysical descriptions (Shouval, Bear, & Cooper, 2002; Lisman & Zhabotinsky, 2001), our model is a phenomenological one. It does not give an explicit interpretation in terms of biophysical quantities such a Calcium concentration (Shouval et al., 2002), CaMKII (Lisman & Zhabotinsky, 2001), glutamate binding, NMDA receptors etc. Rather it aims at a minimal description of the major phenomena observed in electrophysiology experiments.

The advantage of such a minimal model is that it allows us to discuss functional consequences in small (Song & Abbott, 2001; Lubenov & Siapas, 2008; N. Levy, Horn, Meilijson, & Ruppin, 2001), and possibly even large (Morrison, Aertsen, & Diesmann, 2007; Izhikevich & Edelman, 2008) networks. We show that in small networks of up to 10 neurons the learning rule leads to input specificity, necessary for receptive field development – similar to earlier models of STDP (Gerstner et al., 1996; Song & Abbott, 2001) or rate-based plasticity rules (Cooper, Intrator, Blais, & Shouval, 2004; Miller, 1994). Going significantly beyond earlier studies, we explicitly address the question of whether functional connectivity patterns of cortical pyramidal neurons measured in recent electrophysiological studies (Song et al., 2005; Lefort et al., 2009) could be the result of plasticity during continued stimulation of neuronal model networks. We find that connectivity patterns strongly depend on the underlying coding hypothesis: With a temporal coding hypothesis, where input spikes arrive in a fixed temporal order, the recurrent network develops a connectivity pattern with a few strong unidirectional connections. However, under a rate coding paradigm, where stimuli are stationary during a few hundred milliseconds the same network exhibits sustained and strong bidirectional connections. This is in striking contrast to standard STDP rules where bidirectional connections are impossible (Song & Abbott, 2001).

The mathematical simplicity of the model enables us to identify conditions under which it becomes equivalent to the well-known Bienenstock-Cooper-Munro model (Cooper et al., 2004) used in classical rate-based descriptions of developmental learning; and equivalent to some earlier models of STDP (Pfister & Gerstner, 2006) — and why our model is fundamentally different from classical STDP models (Gerstner et al., 1996; Song & Abbott, 2001; Gerstner & Kistler, 2002), widely used for temporal coding.

2.2 Results

In order to study how connectivity patterns in cortex can emerge from an interplay of plasticity rules and coding, we need a plasticity rule that is consistent with a large body of experiments, not just a single paradigm such as STDP. Since synaptic depression and potentiation take place through different pathways (O'Connor, Wittenberg, & Wang, 2005) our model uses separate additive contributions to the plasticity rule, one for LTD and another one for LTP (see Fig. 2.1 and methods).

2.2.1 Fitting the Plasticity Model to Experimental Data

Consistent with voltage clamp (Ngezahayo et al., 2000) and stationary depolarization experiments (Artola et al., 1990), LTD is triggered in our model if presynaptic spike arrival occurs while the membrane potential of the postsynaptic neuron is slightly depolarized (above a threshold θ_- usually set to resting potential) whereas LTP occurs if depolarization is big (above a second threshold θ_+, see Fig. 2.1). The mathematical formulation of the plasticity rule makes a distinction between the momentary voltage u and the low-pass filtered voltage variables $\bar{u}_-$ or $\bar{u}_+$ which denote temporal averages

of the voltage over the recent past (the symbols $\bar{u}_-$ and $\bar{u}_+$ indicate filtering of u with two different time constants). Similarly, the event x of presynaptic spike arrival needs to be distinguished from the trace $\bar{x}$ that is left at the synapse after stimulation by neurotransmitter. Potentiation occurs only if the momentary voltage is above θ_+ (this condition is fulfilled during action potential firing) AND the average voltage $\bar{u}_+$ above θ_- (this is fulfilled if there was a depolarization in the recent past) AND the trace $\bar{x}$ left by a previous presynaptic spike event is nonzero (this condition holds if a presynaptic spike arrived a few milliseconds earlier at the synapse); these conditions for plasticity are illustrated in Fig. 2.1 B. LTD occurs if the average voltage $\bar{u}_-$ is above θ_- at the moment of a presynaptic spike arrival (see Fig. 2.1 A). The amount of LTD in our model depends on a homeostatic process on a slower time scale (Turrigiano & Nelson, 2004). Low-pass filtering of the voltage by the variable ($\bar{u}_-$ or $\bar{u}_+$) refers to some unidentified intracellular processes triggered by depolarization, e.g., increase in calcium concentration or second messenger chains. Similarly, the biophysical nature of the trace $\bar{x}$ is irrelevant for the functionality of the model, but a good candidate process is the fraction of glutamate bound to postsynaptic receptors.

We checked the performance of the model on a simulated STDP protocol, where presynaptic spikes arrive a few milliseconds before or after a postsynaptic spike that is triggered by a strong depolarizing current pulse. If a post-pre pairing with a timing difference of 10 ms is repeated 60 times at frequencies below 35 Hz, LTD occurs in our model (Fig. 2.2 A, B), consistent with experiments (Sjöström et al., 2001). Repeated pre-post pairings (with 10 ms timing difference) at frequencies above 10 Hz yield LTP, but pairings at 0.1 Hz do not show any significant change in the model or in experiments (Sjöström et al., 2001). In the model these results can be explained by the fact that at 0.1 Hz repetition frequency, the low-pass filtered voltage $\bar{u}_+$ which increases abruptly during postsynaptic spiking decays back to zero before the next impulse arrives, so that LTP can not be triggered. However, since LTD in the model requires only a weak depolarization of $\bar{u}_-$ at the moment of presynaptic spike arrival, post-pre pairings give rise to depression, even at very low frequency. At repetition frequencies of 50 Hz, the post-pre paradigm is nearly indistinguishable from a pre-post timing, and LTP dominates.

Since spike timing dependence in our model is induced only indirectly via voltage dependence of the model, we wondered whether our model would also be able to account for the intricate interactions of voltage and spike timing reported in (Sjöström et al., 2001). If a pre-post protocol at 0.1 Hz, that normally does not induce LTP, is combined with a depolarizing current pulse (lasting from 50 ms before to 50 ms after the postsynaptic firing event), then potentiation is observed in the experiments (Sjöström et al., 2001), as well as in our model (Fig. 2.2 C, F, I). Due to the injected current, the low-pass filtered voltage variable $\bar{u}_+$ is depolarized before the pairing. Thus at the moment of the postsynaptic spike, the average voltage $\bar{u}_+$ is above the threshold θ_-, leading to potentiation. Similarly, a pre-post protocol that normally leads to LTP can be blocked if the postsynaptic spikes are triggered on the background of a hyperpolarizing current (Fig. 2.2 E, H, I).

In order to study some nonlinear aspects of STDP, we simulate a protocol of burst

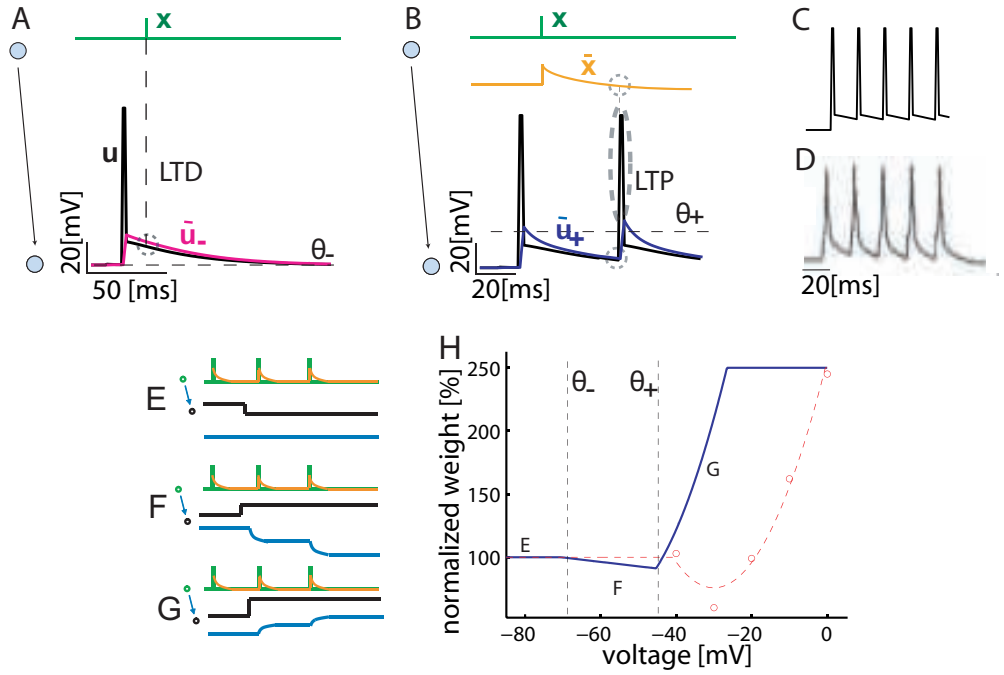


Figure 2.1: Illustration of the model. Synaptic weights react to presynaptic events (top) and postsynaptic membrane potential (bottom) A: The synaptic weight is decreased if a presynaptic spike x (green) arrives when the low pass filtered value $\bar{u}_-$ (magenta) of the membrane potential is above θ_- (dashed horizontal line) B: The synaptic weight is increased if the membrane potential u (black) is above a threshold θ^+ and the low pass filtered value of the membrane potential $\bar{u}_+$ (blue) higher than a threshold θ^- as well as the presynaptic low pass filter $\bar{x}$ (orange) non zero. C: Step current injection makes the postsynaptic neuron fire at 50 Hz in the absence of presynaptic stimulation (membrane potential u in black). No weight change is observed. Note the depolarizing spike-afterpotential consistent with experimental data shown in panel D, reproduced from (Sjöström et al., 2001). E-H: Voltage clamp experiment. A neuron receives weak presynaptic stimulation of 2 Hz during 50 s while the postsynaptic voltage is clamped to values between −60 mV and 0 mV. E-G: Schematic drawing of the trace $\bar{x}$ (orange) of the presynaptic spike train (green) as well as the voltage (black) and the synaptic weight (blue) for the experimental conditions of hyperpolarization (panel E), slight depolarization (panel F) and large depolarization (panel G). H: The weight change as a function of clamped voltage using the standard set of parameters for visual cortex data (blue line, voltage paired with 25 spikes at the synapse). With a different set of parameters the model fits experimental data (red circles) in hippocampal slices (Ngezahayo et al., 2000), see methods for details.

timing-dependent plasticity where presynaptic spikes are paired with one, two or three postsynaptic spikes (Nevian & Sakmann, 2006) (see B). We observe that 60 pre-post pairs at 0.1 Hz do not change the synaptic weight, as discussed above. However, repeated triplets of the form pre-post-post generate potentiation in our model because the first postsynaptic spike induces a depolarizing spike after potential so that $\bar{u}_+$ is

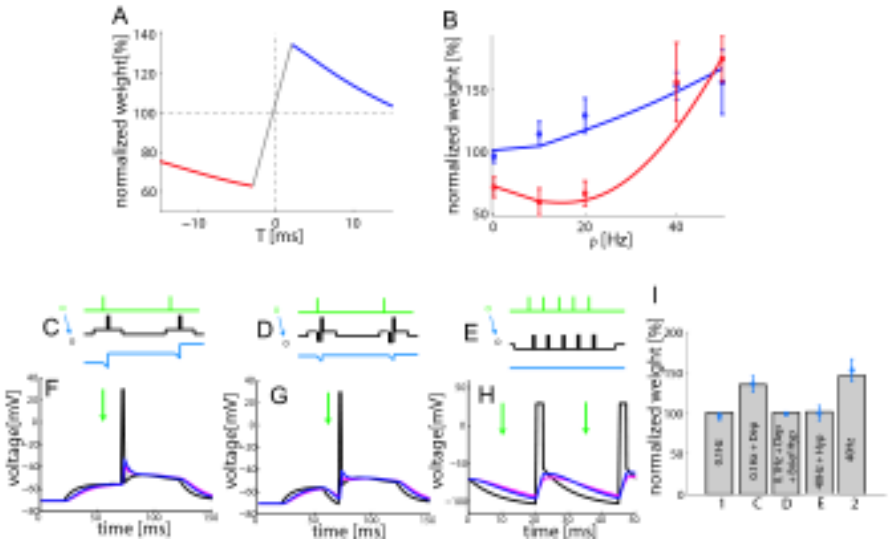


Figure 2.2: Fitting the model to experimental data. A-B: Simulated STDP experiments. A: Spike timing-dependent learning window. The change of the synaptic weight is shown for different time intervals T between the presynaptic and the postsynaptic spike using 60 presynaptic/postsynaptic spike pairs at 20 Hz. B: Weight change as a function of repetition frequency for five spike pairs at frequency ρ with a time delay of +10 ms (pre-post, blue) and -10 ms (post-pre, red), repeated 15 times at 0.1 Hz (only 10 times for frequency of 0.1 Hz). Weight changes are shown as a function of the frequency, dots represent the data taken from (Sjöström et al., 2001) and lines the plasticity model simulation. C-I: Interaction of voltage and STDP. C-E: Schematic induction protocols (green: presynaptic input, black: postsynaptic current, blue: evolution of synaptic weight). C: Low-Frequency Potentiation is rescued by depolarization, see (Sjöström et al., 2001). Low frequency (0.1 Hz) pre-post spike pairs yield LTP if a 100 ms long depolarized current is injected around the pairing. D: LTP fails in the previous scenario if an additional brief hyperpolarized pulse is applied 14 ms before postsynaptic spike so that voltage is brought to rest. E: Hyperpolarization preceding action potential prevents potentiation. In (Sjöström et al., 2001) it is shown that high frequency (40 Hz) pairing leads to LTP. However, when a constant hyperpolarizing current is applied on top of the short pulses inducing the spikes, no weight change is measured. F: The simulated postsynaptic voltage u (black), following protocol of panel C ,is shown as well as the temporal averages $\bar{u}_-$ (magenta) and $\bar{u}_+$ (blue). The presynaptic spike time is indicated by the green arrow. Using the model (B.3) this setting results in potentiation. G: Same as F, but following protocol D. No weight change is measured. H: Same as F, but following protocol E. No weight change is measured. I: Histogram summarizing the normalized synaptic weight of the simulation (bar) and the experimental data (Sjöström et al., 2001) (dot, blue bar=variance) 0.1 Hz pairing (control 1); 0.1 Hz pairing with the depolarization (protocol C); 0.1 Hz pairing with the depolarization and brief hyperpolarization (protocol D); 40 Hz pairing (control 2); 40 Hz pairing with the constant hyperpolarization (protocol E). The parameters are summarized in Table B.2.

depolarized. Adding a third postsynaptic spike to the protocol (yielding quadruplets pre-post-post-post) does not lead to stronger LTP (Fig. 2.3A). Our model also describes the dependence of LTP upon the intra-burst frequency (Fig. 2.3B). At an intra-burst frequency of 20 Hz, no LTP occurs, because the second spike in the burst comes so late that the presynaptic trace $\bar{x}$ has decayed back to zero. At higher intra-burst frequencies, the three conditions for LTP ($u(t) > \theta_+$ and $\bar{u}_+ > \theta_-$ and $\bar{x} > 0$) are fulfilled. The burst timing dependence (Fig. 2.3C) where the timing of one presynaptic spike is changed with respect to a burst of three postsynaptic spikes is qualitatively similar to that found in experiments (Nevian & Sakmann, 2006), but only four of the six experimental data points are quantitatively reproduced by the model with a given set of parameters. Since parameter search does not give a unique result, the prediction of the model has some uncertainty so that three different curves of burst timing-dependent plasticity, corresponding to three equally good choices of parameters have been plotted. Interestingly, our model predicts that the curve of burst timing-dependent plasticity should show a significant change in the amount of potentiation whenever the presynaptic spike is shifted across one of the three postsynaptic spikes (Fig. 2.3C).

2.2.2 Functional Implications

Connectivity patterns in a local cortical circuit have been shown to be non-random, i.e. the majority of connections are weak and the rare strong ones have a high probability of being bidirectional (Song et al., 2005). However, standard models of STDP (Gerstner & Kistler, 2002) do not exhibit stable bidirectional connections (N. Levy et al., 2001; Kozloski & Cecchi, 2008). Intuitively, if the cell A fires before the cell B, a pre-post pairing for the 'AB' connection is formed so that the connection is strengthened. The post-pre pairing occurring at the same time in the 'BA' connection leads to depression. Therefore it is impossible to strengthen both connections at the same time. Moreover, in order to assure long-term stability of firing rates, parameters in standard STDP rules are typically chosen such that inhibition slightly dominates excitation (Gerstner et al., 1996) which implies that under purely random spike firing connections decrease, rather than increase. However, the non-linear aspects of plasticity in our model change such a simple picture. If we simulate two neurons with bidirectional connections at low firing rates, the plasticity model behaves like standard STDP and only unidirectional connections emerge. However, from Fig. 2.2 B and 2.3 B we expect that at higher neuronal firing rates, our model could develop a stable bidirectional connection, in striking contrast to standard STDP rules.

We first simulated a small network of ten all-to-all connected neurons where each neuron fires at a fixed frequency, but the frequencies vary across neurons. We found that bidirectional connections are formed only between pairs of neurons that both fire at high rates, not if one or both of the neurons fire at low frequencies (Fig. 2.4A). In a second simulation, the neurons in the same network are stimulated cyclically such that they are firing in a distinct temporal order (1, 2, 3,...) mimicking an extreme form of temporal coding (Jadhav, Wolfe, & Feldman, 2009). After learning the weights form a loop where strong connections from 1 to 2, 2 to 3, ... develop, whereas bidirectional con-

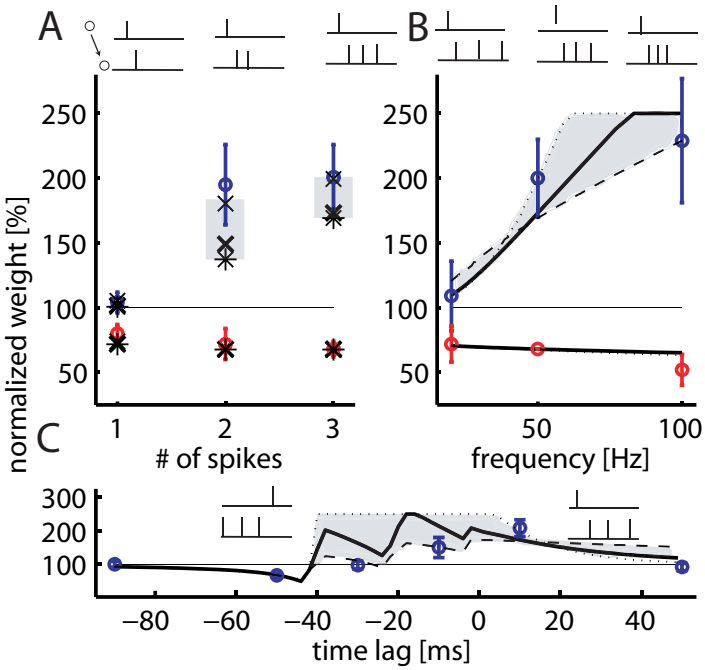


Figure 2.3: Burst timing-dependent plasticity. One presynaptic spike is paired with a burst of postsynaptic spikes. This pairing is repeated 60 times at 0.1 Hz. A: Normalized weight is shown as a function of the number of postsynaptic spikes at 50 Hz. (dots are data from (Nevian & Sakmann, 2006), crosses denote simulation results). The presynaptic spike is paired +10 ms before the first postsynaptic spike (blue) or -10 ms after (red). B: Normalized weight as a function of the frequency between the three postsynaptic action potentials (dots indicate data, lines indicate simulation results and blue denotes pre-post whereas red denotes post-pre pairs). C: Normalized weight as a function of the timing between the presynaptic spike and the first postsynaptic spike of a burst consisting of three spikes at 50 Hz (dot denotes data and black lines simulation results). A hard upper bound has been set to 250% normalized weight. The dashed line and the dotted line represent simulations with different two alternative sets of parameters $A_{\mathrm{LTD}} = 21e^{-5}\,\mathrm{mV}^{-1}$, $A_{\mathrm{LTP}} = 50e^{-4}\,\mathrm{mV}^{-2}$, $\tau_x = 143\,\mathrm{ms}$, $\tau_- = 6\,\mathrm{ms}$, $\tau_+ = 5\,\mathrm{ms}$ and $A_{\mathrm{LTD}} = 21e^{-5}\,\mathrm{mV}^{-1}$, $A_{\mathrm{LTP}} = 67e^{-4}\,\mathrm{mV}^{-2}$, $\tau_x = 5\,\mathrm{ms}$, $\tau_- = 8\,\mathrm{ms}$, $\tau_+ = 5\,\mathrm{ms}$ respectively.

nections (Fig. 2.4B) are depressed. These results are in striking contrast to simulation experiments with a standard STDP rule, where connections are always unidirectional, independently of the stimulation paradigm (Fig. 2.4C, D). Theoretical arguments (see Appendix B) show that bidirectional connections cannot exist under the cyclic temporal stimulation paradigm (neither for standard STDP nor for our plasticity model). Bidirectional connections do develop in our nonlinear voltage dependent plasticity model under the assumption of slowly varying rates – in contrast to standard STDP. (Fig. 2.4 C, D).

We wondered whether the same results would emerge in a more realistic network of excitatory and inhibitory neurons driven by feedforward input. We simulated a network of ten excitatory neurons with all-to-all connectivity and three additional inhibitory neurons. Each inhibitory neuron receives input from eight randomly selected excitatory neurons and randomly projects back to six excitatory neurons. In addition to the recurrent input, each excitatory and inhibitory neuron receives feedforward spike input from 500 presynaptic neurons j modeled as stochastic Poisson spike trains at a rate ν_j. The rates of neighboring input neurons are correlated, mimicking the presence or absence of spatially extended objects. In a rate-coding scheme, the location of the stimulus is switched every 100 ms to a new random position. In case of retinal input, this would correspond to a situation where the subject fixates every 100 ms on a new stationary stimulus. Depending on the retinal position of stimulus, a given postsynaptic neuron responds with low, medium, or high firing rate which is stationary during the 100 ms stimulation period; ie. the firing rates of the ten neurons in the network encode the current position of the stimulus. In a temporal-coding paradigm, the model input is shifted every 20 ms to a neighboring location, mimicking rapid movement of an object across an array of sensory receptors. In this scenario, a given model neuron exhibits only short transient bursts of a few spikes so that it is the temporal structure of the activity (as opposed to stationary firing rates) that encode the position and movement of the stimulus. For both scenarios the network is identical. Feedforward connections and lateral connections between model pyramidal neurons are plastic whereas connections to and from inhibitory neurons are fixed.

During the first 100-400 s of stimulation with the rate-coding paradigm, the excitatory neurons develop localized receptive fields, i.e., weights from *neighboring* inputs to the same postsynaptic neuron become either strong or weak *together* and stay stable thereafter (Fig. 2.5 A). Similarly, lateral connections onto the same postsynaptic neuron develop either exclusively strong or exclusively weak synapses, which remain (apart from fluctuations) stable thereafter (Fig. 2.5 A) leading to a structured pattern of synaptic connections (Fig. 2.5 B). While the labeling of the excitatory neurons at the beginning of the experiment was randomly assigned, we can relabel the neurons after the formation of lateral connectivity patterns so that neurons with similar receptive fields have similar indices, reflecting the neighborhood relation of the network topology. After reordering we can clearly distinguish that three groups of neurons have been formed, characterized by similar receptive fields and strong bidirectional connectivity within the group, and different receptive fields and no lateral connectivity between groups (Fig. 2.5 C). If the overall amplitude of weight change (the "learning rate") is small (compared to

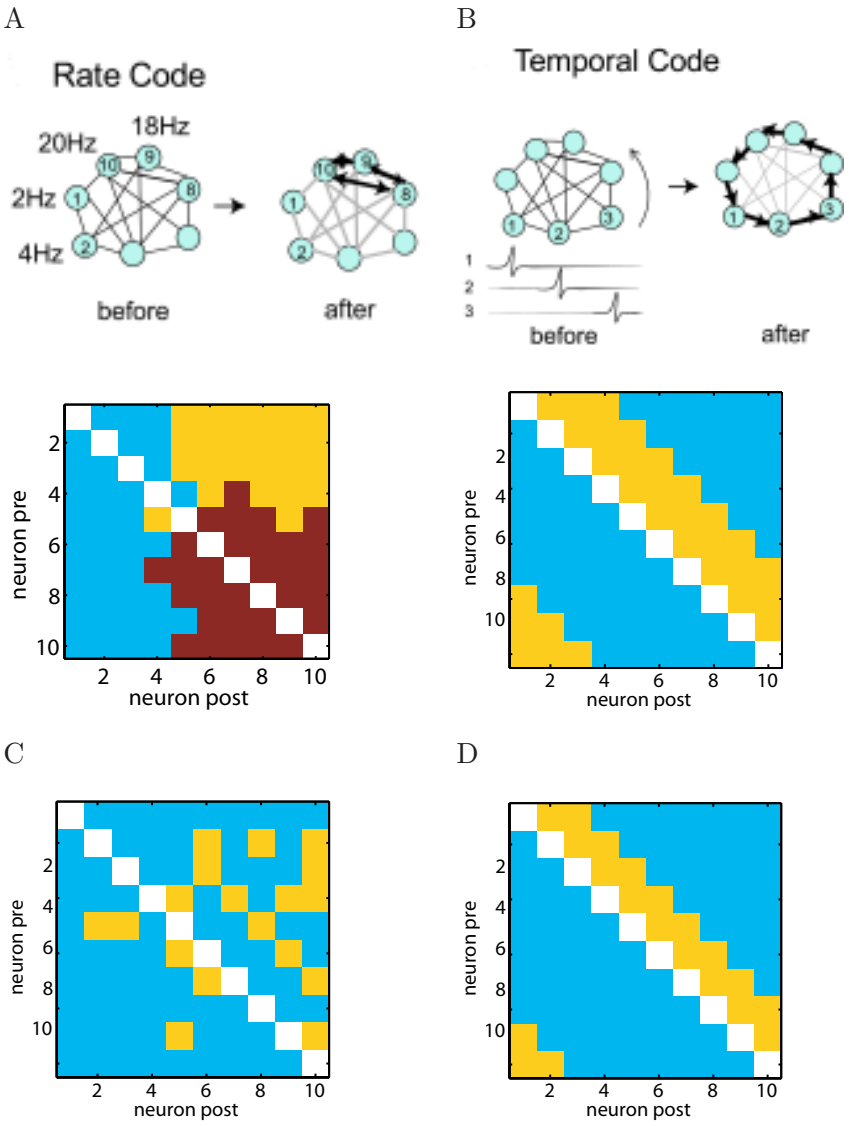


Figure 2.4: Weight evolution in a all-to-all connected network of 10 neurons. A: Rate code: Neurons fire at different frequencies, neuron 1 at 2 Hz, neuron 2 at 4 Hz up to neuron 10 firing at 20 Hz. The weights (bottom) averaged over 100 s show that neurons with high firing rates develop strong bidirectional connections (light blue: weak connections, ie. below 2/3 of the maximal value; yellow: strong unidirectional connections, ie. above 2/3 of the maximal value; brown: strong bidirectional connections). The cluster is schematically represented on top ("after"). B: Temporal code: Neurons fire successively every 20 ms (neuron 1 then 20 ms later neuron 2,...). Connections (bottom) are unidirectional with strong connections from presynaptic neuron with index n (vertical axis) to postsynaptic neuron with index n+1, n+2 and n+3 leading to a ring-like topology (top: schematic). C-D: Same but with standard STDP rule (Gerstner et al., 1996; Song & Abbott, 2001; Gerstner & Kistler, 2002). Bidirectional connections are impossible.

that found in the experiments), the pattern of lateral connectivity is stable and shows a few strong bidirectional connections in a sea of weak lateral connectivity. The reason is that two neurons with similar receptive fields are both active at high rate whenever the stimulus is in the center region of their receptive field. Similar to the simplified model in Fig. 2.4 A, our plasticity rule then gives rise to strong bidirectional lateral connections. Unidirectional strong connections are nearly absent (Fig. 2.5 C and D). If the amplitude and rate of plasticity is more realistic and in agreement with the data of Fig. 2.2, then the pattern of lateral connectivity changes between one snapshot and another one 5 s later, but the overall pattern is stable when averaged over 100 s (Fig. 2.6 A-C). In each snapshot, about half of the strong connections are bidirectional (Fig. 2.6 C and D).

This connectivity pattern is in striking contrast with that shown under a temporal coding paradigm (Fig. 2.7). Neurons develop receptive fields similar to those seen with the rate-coding paradigm, but as expected for temporal Hebbian learning (Gerstner & Kistler, 2002) the receptive field shifts over time (Fig. 2.7 A). With reduced learning rate this shift is slow, as in previous models (Gerstner & Kistler, 2002; Guyonneau et al., 2005) but with realistic learning parameters extracted from the experiments in Fig. 2.2, the shift of the receptive field is surprisingly rapid. More importantly, amongst the lateral connections, strong reciprocal links are nearly absent, whereas strong unidirectional connections from neuron n to neuron $n+1, n+2, n+3$ dominate (Fig. 2.7 B-E). As the feedforward connections (forming the receptive fields) change, the structure of lateral connections changes as well on the time scale of ten minutes. Nevertheless, at each moment in time, the pattern of lateral connections is highly asymmetric, favoring connections from neuron n to $n+k$ (with $k = 1, 2, 3$) over those from n to $n-k$, where n is the neuronal index after relabeling according the receptive field position (Fig. 2.7 A). This suggests that temporal coding paradigms where stimuli are non-stationary and exhibit systematic spatio-temporal correlations, are reflected in the functional connectivity pattern by strong uni-directional connections whereas rate coding (characterized by stationary input with spatial correlations only) leads to strong bidirectional connections. To further explore the relation between pattern and connectivity, we systematically varied both the stimulus duration (between 20 and 100 ms) and the length of the cyclic stimulation sequence (where sequence length one corresponds that a shift to the neighboring location in the cycle is not preferred over a jump to an arbitrary other position and infinity corresponds that the stimulation cycle runs forever without interruption). Our simulation results suggest that a large number of bidirectional connections are only possible if the input does not induce systematic spatio-temporal correlations. Moreover, the amount of asymmetry in connections increases with cycle length, and decreases with the duration of pattern presentation. Bidirectional connections are possible even if the stimulation time is a short as 20 ms, but only if the stimulus has no temporal structure.

2.2.3 Development of Localized Receptive Fields

The results regarding the feedforward connectivity in the previous section lead to the question of the behavior of our plasticity model under stimulation paradigms previously used for rate models (Cooper et al., 2004; Blais, Shouval, & Cooper, 1998; Olshausen

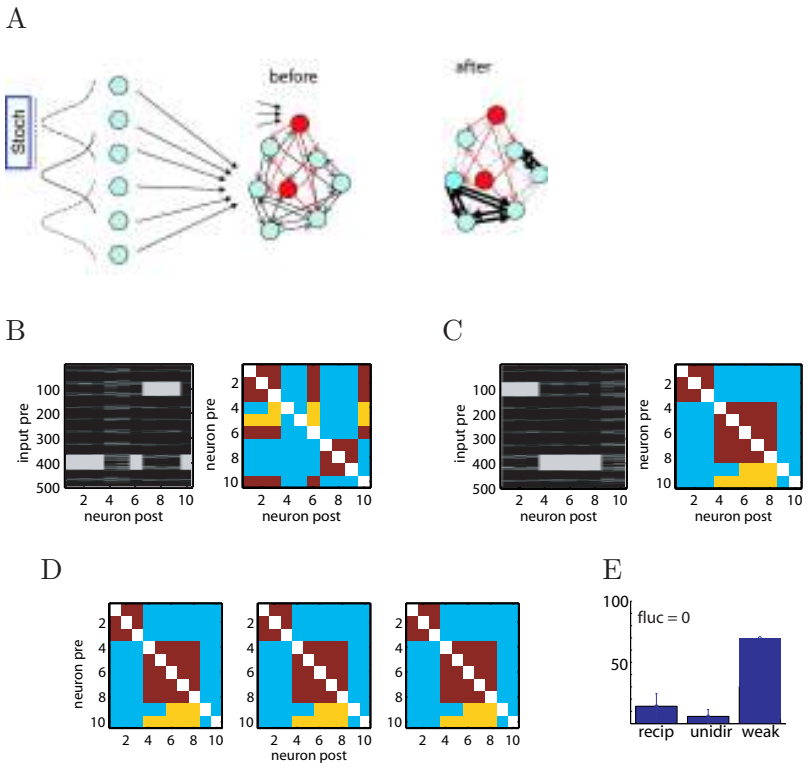


Figure 2.5: Plasticity with a rate coded input and small learning amplitudes. A: A network of ten excitatory neurons (light blue) is connected to three inhibitory neurons (red) and receives feedforward inputs modeled as 500 Poisson spike trains with a Gaussian profile of firing rates. The center of the Gaussian is shifted randomly every 100 ms. The schematic figure shows the network before (left) and after the plasticity experiment (right). B-E: . Model parameters are taken from Table B.2 (visual cortex data) except for the amplitudes A_{LTP} and A_{LTD} which are reduced by a factor 100. B: Mean feedforward weights (left) and recurrent excitatory weights (right) averaged over 100 s. The grey level graph for the feedforward weights (left) indicates that neurons develop receptive fields that are localized in the input space. The recurrent weights (right) are classified as weak (light blue, less than 2/3 of the maximal weight), strong unidirectional (yellow, more than 2/3 of the maximal weight) and strong reciprocal connections (brown). C: Same as B but for the sake of visual clarity the neuron indices are reordered such that neurons with similar receptive fields have adjacent numbers, highlighting that neurons with similar receptive fields (e.g. neurons 1 to 4) have strong bilateral connections. D: Three snap shots of the recurrent connections taken 5 s apart indicating that recurrent connections are stable. E: Histogram of reciprocal, unidirectional and weak connections in the recurrent network averaged over 100 s as in B. The total number of weight fluctuations during 100 s is zero (noted on the figure). The histogram shows an average of ten repetitions (errorbars are the standard deviation).

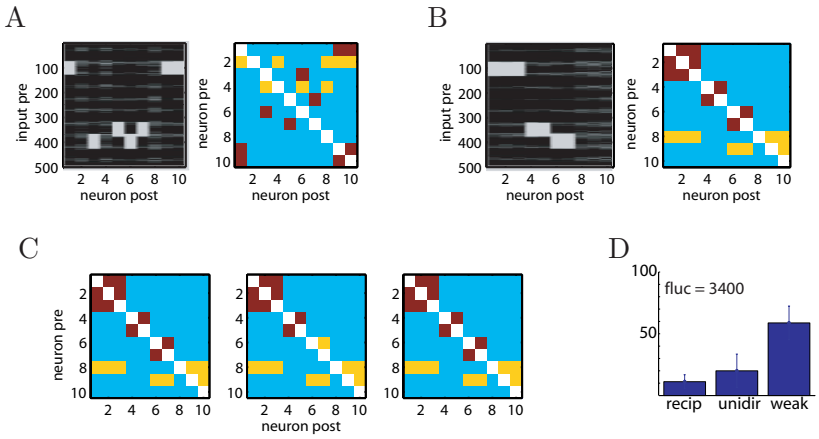


Figure 2.6: Plasticity with a rate coded input and normal learning amplitudes (Table B.2, visual cortex). This figure is similar to Fig. 2.5. A: Receptive fields are localized. B: Reordering allows to visualize that the strong bidirectional give rise to clusters of neurons. These clusters are stable when averaged over 100 s, but connections can change from one time step to the next (see panel C). D: The percentage of reciprocal connections is high, but because of fluctuations more than 1000 transitions between strong unidirectional to strong bidirectional or back occur during 100 s.

& Field, 1996). For rate coding (where pre- and postsynaptic neurons fire with Poisson firing statistics), our plasticity rule presents structural similarities (see Appendix B) to the so-called BCM model presented (Cooper et al., 2004). Both our spiking rule and the rate-based BCM model require presynaptic activity in order to induce plasticity. Furthermore, for our rule as well as for the simplest BCM rule (see (Cooper et al., 2004)), the depression terms are linear and the potentiation terms are quadratic in the postsynaptic variables (i.e. the postsynaptic potential or the postsynaptic firing rate). Beyond these qualitative similarities, an approximate quantitative relation between the BCM model and our model can be constructed under appropriate assumptions. In this case the total weight change Δw in our model is proportional to $\nu^{\text{pre}}\nu^{\text{post}}(\nu^{\text{post}}-\vartheta)$ where ν^{pre} and ν^{post} denote the firing rates of the pre- and postsynaptic neurons respectively and ϑ is a sliding threshold related to the ratio between the LTP and LTD inducing processes (see Appendix B). The sliding threshold arises in our plasticity model because the amount of LTD A_{LTD} depends on the long-term average $\bar{\bar{u}}$ of the voltage on the slow time scale of homeostatic processes. If the amount of LTD increases because of high values of the average voltage, then the threshold ϑ in the above equation (which depends on the ratio of LTP to LTD A_{LTD}/A_{LTP}) increases as well.

Due to its similarities to the BCM model, it is not surprising that our spike-based learning rule with sliding threshold is able to support the development of localized receptive fields, a feature related to independent component analysis (ICA) and sparse coding, see (Cooper et al., 2004; Blais et al., 1998). In our experiments, the input consists of small patches of natural images using standard preprocessing (Olshausen & Field, 1996). Image patches are selected randomly and presented to the neuron for

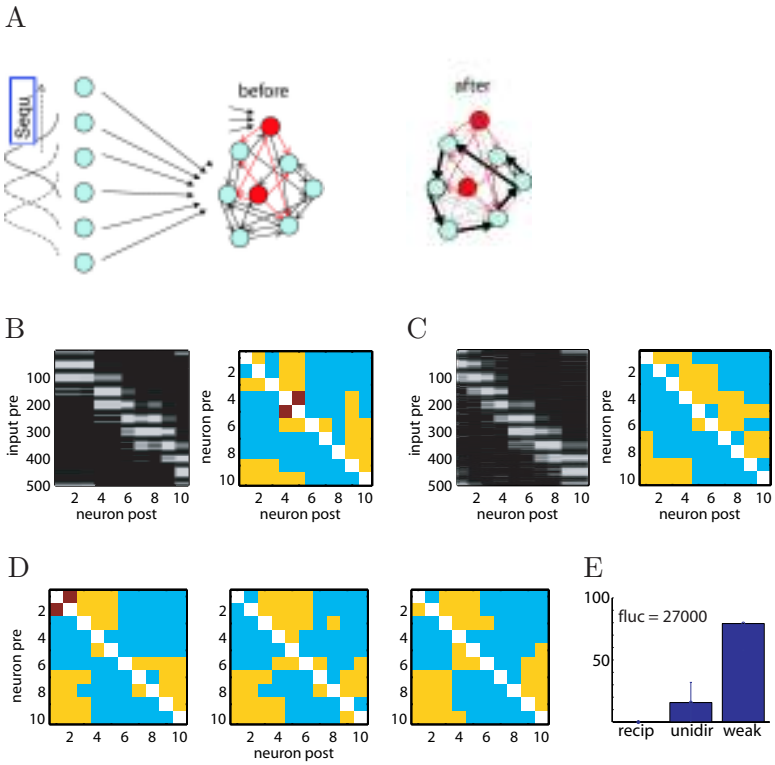


Figure 2.7: Plasticity with input that obeys a temporal coding. The setting is the same as in Fig. 2.5 (parameters from Table B.2, visual cortex) but the input patterns are moved successively every 20 ms, corresponding to a step-wise motion of the Gaussian stimulus profile across the input neurons. A: The schematic figure shows the network before and after the plasticity experiment. Time evolution of the weights. B: Learning with small plasticity amplitudes, yielding localized receptive fields. In the recurrent network a ring-like structure with strong unidirectional connections from neuron 8 (vertical axis) to neuron 9 and 10 (horizontal axis) forms (for neuron 1 to neuron 2, 4, and 5 analogously etc). C: Same as B, but with normal plasticity amplitudes. D: Some of the strong unilateral connections appear or disappear from one time step to the next, but the ring-like network structure persists, since the lines just above the diagonal are much more populated than the line below the diagonal. E: Reciprocal connections are absent, but unidirectional connections fluctuate several times between 'weak' and 'strong' during 100 s.

$T = 200\,\text{ms}$, which is on the order of the fixation time between saccades. Pixel intensities above an average grey value are converted to spike trains of ON-cells and and those below reference intensity to spikes in OFF-cells, using the relative intensity as the rate of a Poisson process. The spike trains from ON- and OFF-cells are the input to a cortical model neuron. The synaptic weights undergo plasticity following our learning rule (B.3). After learning, the weights exhibit a stable spatial structure that can be interpreted as a receptive field (see Fig. 2.8). In contrast to principal component analysis of the image patches (as for example implemented by Hebbian learning in linear neurons (Oja, 1982)), the receptive fields are *localized* (i.e. the region with significant weights does not stretch across the whole image patch). Nine runs of the learning experiments give receptive fields with different locations and orientations (Fig. 2.8 D). Because of the homeostatic control of LTD in our plasticity model, the neuron compensates for increased input firing rates by developing smaller receptive fields that are even more localized (Fig. 2.8 E). Development of localized receptive fields has been interpreted as a signature of ICA or sparse coding (Olshausen & Field, 1996). In contrast to most other ICA algorithms (Hyvaerinen, Karhunen, & Oja, 2001) our rule is biologically more plausible since it is consistent with a large body of plasticity experiments.

2.3 Discussion

Over the last decades plasticity models have primarily focused on questions of development of receptive fields and cortical maps (Cooper et al., 2004), or memory formation (Hopfield, 1982). Because traditional plasticity rules are rate models, the relation between coding and connectivity could not be studied. In contrast, our plasticity rule is formulated on the level of postsynaptic voltage. Since action potentials present large and narrow voltage peaks, they act as singular events in a voltage rule so that in the presence of a spike our rule turns automatically into a spike timing-dependent rule. Indeed, for spike coding (and in the absence of significant subthreshold voltage manipulations) our plasticity rule behaves like a STDP rule where triplets of spikes with pre-post-post or post-pre-post timing evoke LTP, whereas pairs with post-pre timing evoke LTD.

For a comparison of our model with experiments we have mainly focused on experiments in slices of visual cortex, but some of the results can also be related to work in hippocampus. The model can successfully reproduce the voltage dependence of LTP/LTD seen in experiments under depolarization of the postsynaptic membrane (Artola et al., 1990; Ngezahayo et al., 2000). Furthermore, for classical STDP experiments such as (Bi & Poo, 2001; Sjöström et al., 2001; Wang, Gerkin, Nauen, & Bi, 2005), which have a stimulation protocol unambiguously defined in terms of pre- and postsynaptic spike times, the model gives a timing dependence reminiscent of the typical STDP function (Bi & Poo, 2001). Moreover in contrast to standard STDP rules (reviewed in (Gerstner & Kistler, 2002)), more complicated effects such as the pairing frequency dependence (Sjöström et al., 2001) and burst timing dependence plasticity (Nevian & Sakmann, 2006) are qualitatively described. In addition the rule is expected to reproduce the triplet and quadruplet experiments in hippocampal slices (Wang et al., 2005) (data not shown), because for all STDP protocols the plasticity rule in this chapter is similar to a

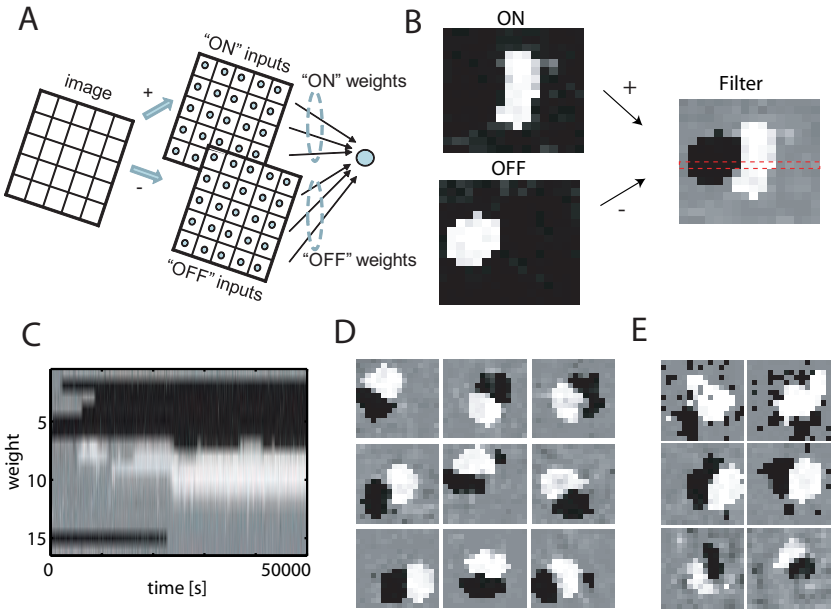


Figure 2.8: Receptive field development. A: A small patch of 16x16 pixels is chosen from the whitened natural images benchmark (Olshausen & Field, 1996). The patch is selected randomly and is presented as input to 512 neurons for 200 ms. The positive part of the image is used as the firing rate to generate Poisson spike trains of the 256 "ON" inputs and the negative one for the 256 "OFF" inputs. B: The weights after convergence are shown for the "ON" inputs and the "OFF" inputs rearranged on a 16x16 image. The filter is calculated by subtracting the "OFF" weights from the "ON" weights. The filter is localized and bimodal, corresponding to an oriented receptive field. C: Temporal evolution of the weights shown in the red dashed box in panel B. D: Nine different neurons. E. Two different neurons receiving presynaptic input with varying firing rates from (top) $0 - 25\,\mathrm{Hz}$ (middle) $0 - 37.5\,\mathrm{Hz}$ (bottom) $0 - 75\,\mathrm{Hz}$.

nonlinear STDP rule presented (Pfister & Gerstner, 2006). Deriving STDP rules from voltage dependence has been attempted before (Sjöström et al., 2001; Saudargiene, Porr, & Wörgötter, 2003; Brader, Senn, & Fusi, 2007). However, since these earlier models use the momentary voltage (Brader et al., 2007) or its derivative (Saudargiene et al., 2003), rather than a combination of momentary and averaged voltage as in our model, they cannot account for the broad range of nonlinear effects in STDP experiments or interaction of voltage and spike timing. The voltage-based model of (Sjöström et al., 2001) which can account for a variety of nonlinear STDP and voltage effects, uses separate empirical functions for timing dependence (width of rectangular STDP window), voltage dependence (sigmoidal function of depolarization just before a spike), frequency dependence (linear dependence), and multiple spike summation with preference for LTP, to capture the nonlinear effects of LTP and its dominance over LTD at higher frequencies. Our model is similar in that it also uses momentary voltage before the spike as one of the variables, but does not require an explicit frequency-dependent term, nor an explicit timing-dependent term. Rather, frequency and timing dependence follow from the model-dynamics. Our model shows similarities with LTP induction in the TagTriC model (Clopath, Ziegler, Vasilaki, Buesing, & Gerstner, 2008), but the TagTriC model focuses on the long-term stability of synapses, rather than spike timing dependence of the induction mechanism.

Even though our model does not require a biophysical interpretation of the variables, it is tempting to speculate about potential mechanisms. For the *depression* term in our model, a 'trace' $\bar{u}_-$ left by previous activity of the postsynaptic neuron is combined with spike arrival x at the presynaptic terminal (Fig. 2.1 A). In view of the results on LTD in layer-V neocortical neurons (Sjöström, Turrigiano, & Nelson, 2003), this trace could be related to endocannaiboids released from the postsynaptic terminal which activate presynaptic CB1 receptors. Coincidence of this slow trace with the activation of presynaptic NMDA receptors (which rapidly respond to the glutamate released by presynaptic activity $x(t)$) could be the trigger signal for LTD (Sjöström et al., 2003). Indeed, the duration of the LTD part in the STDP function increases, if the endocannabinoid trace is artificially prolonged (see Fig 9 of (Sjöström et al., 2003)). In other neuron types and brain areas, the same mathematical model (but with different parameters) could correspond to different biophysical mechanisms of LTD. For example, in hippocampal CA1 neurons, the trace $\bar{u}_-$ could reflect the calcium entry through voltage-gated ion channels during depolarization which, when combined with synaptic signals (caused by the presynaptic spike arrival x), would give rise to the modest calcium signals necessary to trigger LTD (reviewed in (Lisman & Zhabotinsky, 2001; Shouval et al., 2002; Sjöström et al., 2003)). *Potentiation* is induced in our model by the combination of three factors: a momentary as well as an "averaged" postsynptic depolarization and the presence of a 'trace' $\bar{x}$ left by presynaptic spike arrival (Fig. 2.1 A). The trace $\bar{x}$ could correspond to the amount of glutamate bound to the postsynaptic NMDA receptor even though it has been argued that the time constant of unbinding would be too long compared to the duration of the LTP part of the STDP function (Sjöström et al., 2003). A high momentary voltage u can be induced by a backpropagating action potential. Interestingly, backpropagation of action potentials is more likely and more reliable to occur in

the background of a weak depolarization of the dendrite (reviewed in (Sjöström et al., 2003)) – and such a weak depolarization potentially corresponds the the term $\bar{u}_+$ in our model. Because in our model we have a depolarizing afterpotential after each spike (Fig. 2.1 C, similar to that seen in experiments Fig. 2.1 D), the value of $\bar{u}$ just before the next spike increases with the repetition frequency of the STDP protocol, in agreement with experiments (Fig. 5 D in (Sjöström et al., 2001)). Our model is therefore consistent with results that LTP can be induced in distal synapses only if additional cooperative input or dendritic depolarization prevent failure of backpropagating action potentials (Sjöström & Häusser, 2006). In the context of the classical view of the NMDA receptor as a coincidence detector (reviewed in (Sjöström et al., 2003)), it is quite natural to see why a sequence post-pre-post of two postsynaptic action potentials and one presynaptic spike are ideal for LTP: The spike-afterpotential of the first postsynaptic action potential removes the calcium block and prepares the dendrite for successful backpropagation of a later action potential. If the backpropagating action potential caused by the second postsynaptic spike occurs just slightly after presynaptic spike arrival, this causes a sharply peaked and large calcium transient that would be sufficient to trigger the LTP induction chain. We note that a post-pre-post sequence is also the ideal trigger for potentiation in our model. Even though our model is formulated on the level of voltage, we do not imply that voltage itself is the essential biophysical mechanism. Rather, under physiological conditions, the voltage transient (or current or conductance transient) caused by synaptic input or action potential firing is the starting point for long biochemical signaling chains that, in the end lead to the induction of plasticity. In a phenomenological model, the signature of the inputs (here the voltage transients) are directly linked (via mathematical variables or "traces") to the induction of plasticity, jumping over the biophysical mechanisms of the signal transduction chain. For the description of experiments where the experimentalist directly interacts with intermediate steps of the signaling chain (e.g. artificial calcium release or gelator, blocking of specific channels or kinases) a detailed biophysical model is necessary while for a compressed summary of the electrophysiological experiments considered in this chapter a phenomenological model is sufficient.

Our plasticity rule allows us to explain experiments from two different laboratories by one single principle. Both the "potentiation is rescued by depolarization" (Sjöström et al., 2001) scenario (Fig. 2.2 F) and that of burst timing-dependent LTP (Nevian & Sakmann, 2006) (Fig. 2.3) indicate that LTP at low frequency is induced when the membrane is depolarized before the pre-post pairing. This depolarization can be due to a previous spike during a postsynaptic burst (Nevian & Sakmann, 2006) or to a depolarization current. A further unexpected result is that, with the set of parameters derived from visual cortex slice experiments, synapses fluctuate rapidly between strong and weak weights. This aspect is interesting in view of synapse mobility reported in imaging experiments (Yuste & Bonhoeffer, 2004).

Our phenomenological model gives a compressed description of the experimental results discussed above and we believe that it cannot be simplified further. First, voltage is necessary as a variable whenever voltage is manipulated in experiments, and second, dependence upon voltage must be nonlinear to account for the experimental results with

stationary voltage (Artola et al., 1990; Ngezahayo et al., 2000). Phenomenological models have some freedom in the choice of the mathematical form of the nonlinearities (e.g. exponential, polynomial or piecewise linear functions) and we chose a suitable combination of piecewise linear functions with thresholds θ_+ and θ_-. Third, the frequency dependence of STDP as well as the interaction of voltage with spike timing indicates that the temporal relation between stimulation events is important. All timing relations in our phenomenological model have been implemented as (first-order) linear filtering, the simplest method at hand. For the case of classical STDP experiments, where all spikes are triggered by the experimenter our phenomenological model can be simplified and becomes identical or closely related to existing nonlinear STDP models (Pfister & Gerstner, 2006; Senn, Tsodyks, & Markram, 2001), but regarding the interaction between voltage and spike timing such a further simplification is not possible. Finally, the fact that the curve of burst timing-dependent plasticity (Fig. 2.3 C) is not perfectly reproduced indicates that our plasticity model does not have an unnecessarily large number of free parameters.

There are, however, certain limitations to our plasticity rule. First, we did not address the problem of weight dependence of synaptic plasticity and simply assumed that weights can grow to a hard upper bound. Nevertheless, the rule can easily be adapted to include soft bounds (Gerstner & Kistler, 2002) by changing the prefactors $A_{\text{LTP}}, A_{\text{LTD}}$ accordingly (Clopath et al., 2008). Second, short term plasticity (Tsodyks & Markram, 1997) could be added for a better description of the plasticity phenomena occurring especially during high frequency protocols. Third, our plasticity rule describes only induction of potentiation or depression during the early phase of LTP/LTD (Frey & Morris, 1997). Additional mechanisms need to be implemented in the model to describe the transition from early to late LTP/LTD (Clopath et al., 2008; Barrett, Billings, Morris, & Rossum, 2009). Finally, in modeling voltage-clamp experiments, we assume in our model a unique voltage throughout the whole neuron. In particular the dendrite is assumed to be equipotential to the soma. Yet, experiments controlling the voltage at the soma do not guarantee an equal or even fixed voltage at the site of the synapses with respect to the soma. An obvious and promising improvement would be to use a multi-compartment neuron model (e.g. distinct compartments for the soma and dendrites). In the presented work we did not use a more sophisticated multi-compartment model as this would introduce a considerable number of new parameters making overfitting more likely to occur. Interestingly, our voltage-based formulation of plasticity applied locally in a compartmental model would allow potentiation to occur in a dendritic branch whenever the following three conditions are met: presynaptic activity, recent postsynaptic depolarization, and momentary large depolarization – independent of the source of of depolarization. Hence, dendritic spikes could lead to potentiation in the absence of somatic action potentials, in agreement with recent experiments in hippocampal slices (Hardie & Spruston, 2009).

Our plasticity model leads to several predictions that could be tested in slice experiments. First, under the assumption of voltage clamp, our rule is linear in the presynaptic activities (see Appendix B). Thus the model predicts that in voltage clamp experiments the weight change is only dependent on the voltage and the *number* of presynaptic spikes

but not on their exact timing i.e. low frequency, tetanus, burst input should give the same result. Second, in the scenario where potentiation is rescued by depolarization, the amount of weight change should be the same whether a depolarizing current of a certain amplitude stops precisely when the postsynaptic spike is triggered or whether a current of slightly bigger amplitude stops a few milliseconds earlier. Third, multiple STDP experiments have shown that pre-post pairing (with 10 ms timing difference) repeated at 10 Hz leads to potentiation (Sjöström et al., 2001). In our plasticity model, LTP occurs in that case because the depolarizing spike-afterpotential of the last postsynaptic spike leads to an increase of the filtered membrane voltage just before the next postsynaptic spike. If this interpretation is correct, a hyperpolarizing current sufficient to cancel the spike afterpotential during 40 ms should block LTP (note that at 10 Hz repetition frequency this is different from blocking LTP by a hyperpolarizing current a few milliseconds *before* the next spike (Sjöström et al., 2001)).

The influence of STDP on temporal coding has been studied in the past primarily with respect to changes in the feedforward connections (reviewed in (Gerstner & Kistler, 2002)). The effect of STDP on lateral connectivity has been studied much less (N. Levy et al., 2001; Morrison et al., 2007; Izhikevich & Edelman, 2008; Lubenov & Siapas, 2008). We have shown in this chapter that, because of STDP, coding influences the network topology, as different codes give rise to different patterns of lateral connectivity. Our results are in contrast to standard STDP rules which always suppress short loops, and in particular bidirectional connections (N. Levy et al., 2001; Kozloski & Cecchi, 2008). Our more realistic plasticity model shows that under a rate coding paradigm (where the neuron is stimulated by different stationary patterns) bidirectional connectivity and highly connected clusters with multiple loops are not only possible, but even dominant. It is only for temporal coding (characterized by stimulation with significant spatiotemporal correlations), that our biologically plausible rule leads to dominant unilateral directions. We speculate that the differences in coding between different brain areas could lead, even if the learning rule were exactly the same, to different network topologies. Our model predicts that experiments where cells in a recurrent network are repeatedly stimulated in a fixed order would decrease the fraction of strong bidirectional connections, whereas a stimulation pattern where clusters of neurons fire at high rate during episodes of a few hundred milliseconds would increase this fraction. In this views it is tempting to connect the low degree of bidirectional connectivity in barrel cortex (Lefort et al., 2009) to the bigger importance of temporal structure in whisker input (Jadhav et al., 2009), compared to visual input.

2.4 Acknowledgments

This chapter is based on the paper *Connectivity reflects coding: A model of voltage-based STDP with homeostasis*, which was written by Claudia Clopath (CC), Lars Büsing (LB), Eleni Valsilaki (EV) and Wulfram Gerstner (WG). The plasticity model was developped by CC and LB. CC fitted the model to experimental data, designed and carried out the simulations. LB developped the link to the BCM rule and did the calculations for the expected weight change presented in Appendix B. EV participated in discussions. WG

supervised the project and wrote most of the manuscript.

Chapter 3

Simplified Information Bottleneck Optimization with Spiking Neurons

Contents

We show that under suitable assumptions (primarily linearization) a simple and perspicuous online learning rule for Information Bottleneck optimization with spiking neurons can be derived. This rule performs on common benchmark tasks as well as a rather complex rule that has previously been proposed (Klampfl, Legenstein, & Maass, 2009). Furthermore, the transparency of this new learning rule makes a theoretical analysis of its convergence properties feasible. A variation of this learning rule (with sign changes) provides a theoretically founded method for performing Principal Component Analysis (PCA) with spiking neurons. By applying this rule to an ensemble of neurons, different principal components of the input can be extracted. In addition, it is possible to preferentially extract those principal components from incoming signals X that are related or are not related to some additional target signal Y_T. In a biological interpretation, this target signal Y_T (also called relevance variable) could represent proprioceptive feedback, input from other sensory modalities, or top-down signals.

3.1 Introduction

The Information Bottleneck (IB) approach (Tishby, Pereira, & Bialek, 1999) allows the investigation of learning algorithms for unsupervised and semi-supervised learning on the basis of clear optimality principles from information theory. Two types of time-varying inputs X and Y_T are considered. The learning goal is to learn a transformation from X into another signal Y that extracts only those components from X that are related

to the relevance signal Y_T. In a more global biological interpretation X might represent for example some sensory input, and Y the output of the first processing stage for X in the cortex. In this article Y will simply be the spike output of a neuron that receives the spike trains X as inputs. The starting point for our analysis is the first learning rule for IB optimization in for this setup, which has recently been proposed in (Klampfl, Legenstein, & Maass, 2007), (Klampfl et al., 2009). Unfortunately, this learning rule is complicated (and restricted to discrete time), and no theoretical analysis of its behavior is feasible. Any online learning rule for IB optimization has to make a number of simplifying assumptions, since true IB optimization can only be carried out in an offline setting. We show here, that with a slightly different set of assumptions than those made in (Klampfl et al., 2007) and (Klampfl et al., 2009), one arrives at a drastically simpler and intuitively perspicuous online learning rule for IB optimization with spiking neurons. The learning rule in (Klampfl et al., 2007) was derived by maximizing the objective function[1] L_0:

$$L_0 = -I(X,Y) + \beta I(Y,Y_T) - \gamma D_{KL}(P(Y)\|P(\tilde{Y})), \tag{3.1}$$

where $I(.,.)$ denotes the mutual information between its arguments and β is a positive trade-off factor. The target signal was assumed to be given by a spike train Y_T. The learning rule from (Klampfl et al., 2007) (see (Klampfl et al., 2009) for a detailed interpretation) is quite involved and requires numerous auxiliary definitions (hence we cannot repeat it in this abstract). Furthermore, it can only be formulated in discrete time (steps size Δt) for reasons we want to outline briefly: In the limit $\Delta t \to 0$ the essential contribution to the learning rule, which stems from maximizing the mutual information $I(Y,Y_T)$ between output and target signal, vanishes. This difficulty is rooted in a rather technical assumption, made in appendix A.4 in (Klampfl et al., 2009), concerning the expectation value $\overline{\overline{\rho}}^k$ at time step k of the neural firing probability ρ , given the information about the postsynaptic spikes and the target signal spikes up to the preceding time step $k-1$ (see our detailed discussion in the appendix C.5)[2]. The restriction to discrete time prevents the application of powerful analytical methods like the Fokker-Planck equation, which requires continuous time, for analyzing the dynamics of the learning rule.

In section 3.2 of this chapter, we propose a much simpler learning rule for IB optimization with spiking neurons, which can also be formulated in continuous time. In contrast to (Klampfl et al., 2009), we approximate the critical term $\overline{\overline{\rho}}^k$ with a linear estimator, under the assumption that X and Y_T are positively correlated. Further simplifications in comparison to (Klampfl et al., 2009) are achieved by considering a simpler neuron model (the linear Poisson neuron, see (Gerstner & Kistler, 2002)). However we show through computer simulation in the appendix C that the resulting simple learning

[1] The term $D_{KL}(P(Y)\|P(\tilde{Y}))$ denotes the Kullback-Leibler divergence between the distribution $P(Y)$ and a target distribution $P(\tilde{Y})$. This term ensures that the weights remain bounded, it shortly discussed in appendix C.

[2] The remedy, proposed in section 3.1 in (Klampfl et al., 2009), of replacing the mutual information $I(Y,Y_T)$ in L_0 by an information rate $I(Y,Y_T)/\Delta t$ does not solve this problem, as the term $I(Y,Y_T)/\Delta t$ diverges in the continuous time limit.

rule performs equally well for the more complex neuron model with refractoriness from (Klampfl et al., 2007) - (Gerstner & Kistler, 2002). The learning rule presented here can be analyzed by the means of the drift function of the corresponding Fokker-Planck equation. The theoretical results are outlined in section 3.3, followed by the consideration of a concrete IB optimization task in section 3.4. A link between the presented learning rule and Principal Component Analysis (PCA) is established in section 3.5. A more detailed comparison of the learning rule presented here and the one of (Klampfl et al., 2009) as well as results of extensive computer tests on common benchmark tasks can be found in the appendix C.

3.2 Neuron Model and Learning Rule for IB Optimization

We consider a linear Poisson neuron with N synapses of weights $w = (w_1, \ldots, w_N)$. It is driven by the input X, consisting of N spike trains $X_j(t) = \sum_i \delta(t - t_j^i),\ j \in \{1, \ldots, N\}$, where t_j^i denotes the time of the i'th spike at synapse j. The membrane potential $u(t)$ of the neuron at time t is given by the weighted sum of the presynaptic activities $\nu(t) = (\nu_1(t), \ldots, \nu_N(t))$:

$$\begin{aligned} u(t) &= \sum_{j=1}^{N} w_j \nu_j(t) \\ \nu_j(t) &= \int_{-\infty}^{t} \epsilon(t-s) X_j(s) ds. \end{aligned} \tag{3.2}$$

The kernel $\epsilon(.)$ models the EPSP of a single spike (in simulations $\epsilon(t)$ was chosen to be a decaying exponential with a time constant of $\tau_m = 10\,\text{ms}$). The postsynaptic neuron spikes at time t with the probability density $g(t)$:

$$g(t) = \frac{u(t)}{u_0},$$

with u_0 being a normalization constant. The postsynaptic spike train is denoted as $Y(t) = \sum_i \delta(t - t_f^i)$, with the firing times t_f^i.

We now consider the IB task described in general in (Tishby et al., 1999), which consists of maximizing the objective function L_{IB}, in the context of spiking neurons. As in (Toyoizumi, Pfister, Aihara, & Gerstner, 2007), we introduce a further term L_3 into the the objective function that reflects the higher metabolic costs for the neuron to maintain strong synapses, a natural, simple choice being $L_3 = -\lambda \sum w_j^2$. Thus the complete objective function L to maximize is:

$$L = L_{\text{IB}} + L_3 = -I(X,Y) + \beta I(Y_T, Y) - \lambda \sum_{j=1}^{N} w_j^2. \tag{3.3}$$

The objective function L differs slightly from L_0 given in (3.1), which was optimized in (Klampfl et al., 2009); this change turned out to be advantageous for the PCA learning

rule given in section 3.5, without significantly changing the characteristics of the IB learning rule.

The online learning rule governing the change of the weights $w_j(t)$ at time t is obtained by a gradient ascent of the objective function L:

$$\frac{d}{dt}w_j(t) = \alpha\frac{\partial L}{\partial w_j}.$$

For small learning rates α and under the assumption that the presynaptic input X and the target signal Y_T are stationary processes, the following learning rule can be derived:

$$\frac{d}{dt}w_j(t) = \alpha\frac{Y(t)}{u(t)}\nu_j(t)\left(-\frac{u(t)-\overline{u}(t)}{\overline{u}(t)} + \beta\left(F[Y_T](t) - \overline{F[Y_T]}(t)\right)\right) - \alpha\lambda w_j(t), \tag{3.4}$$

where the operator $\overline{(.)}$ denotes the low-pass filter with a time constant τ_C (in simulations $\tau_C = 3s$), i. e. for a function f:

$$\overline{f}(t) = \frac{1}{\tau_C}\int_{-\infty}^{t}\exp\left(-\frac{t-s}{\tau_C}\right)f(s)ds. \tag{3.5}$$

The operator $F[Y_T](t)$ appearing in (3.4) is proportional to the expectation value of the membrane potential $\langle u(t)\rangle_{X|Y_T} = E[u(t)|Y_T]$, given the observations $(Y_T(\tau)|\tau \in \mathbb{R})$ of the relevance signal; F is thus closely linked to estimation and filtering theory. For a known joint distribution of the processes X and Y_T, the operator F could in principal be calculated exactly, but it is not clear how this quantity can be estimated in an online process; thus we look for a simple approximation to F. Under the above assumptions, F is time invariant and can be approximated by a Volterra series (for details see the appendix C):

$$\langle u(t)\rangle_{X|Y_T} \propto F[Y_T](t) = \sum_{n=0}^{\infty}\int_{\mathbb{R}}\cdots\int_{\mathbb{R}}\kappa_n(t-t_1,\ldots,t-t_n)\prod_{i=1}^{n}Y_T(t_i)dt_i. \tag{3.6}$$

In this article, we concentrate on the situation, where F can be well approximated by its linearization, corresponding to a linear estimator of $\langle u(t)\rangle_{X|Y_T}$, which will be called $u_T(t)$:

$$u_T(t) = \int_{\mathbb{R}}\kappa_1(t-t_1)Y_T(t_1)dt_1. \tag{3.7}$$

Assuming positively correlated X and Y_T, $\kappa_1(t)$ is chosen to be a non-anticipating decaying exponential $\exp(-t/\tau_0)\Theta(t)$ with a time constant τ_0 (in simulations $\tau_0 = 100\,\text{ms}$), where $\Theta(t)$ is the Heaviside step function. This choice is motivated by the standard models for the impact of neuromodulators (see (Izhikevich, 2007)), thus such a kernel may be implemented in a realistic biological mechanism. Furthermore, the resulting $u_T(t)$ is the causal Wiener filter[3] of this estimation problem, if the autocorrelation of

[3]The Wiener filter is the optimal causal linear estimator in the mean-square error sense and its kernel κ_1 fulfills the Wiener-Hopf equation (equation 14-88 in (Papoulis, 1991)), for more details see the

Y_T and the cross-correlation function between u and Y_T decay exponentially with the time constant τ_0. It turned out that the choice of τ_0 was not critical, it could be varied over a decade ranging from 10 ms to 100 ms.

Using the above definitions, the resulting learning rule is given by (in vector notation):

$$\frac{d}{dt}w(t) = \alpha\frac{Y(t)}{u(t)}\nu(t)\left(-\frac{u(t)-\overline{u}(t)}{\overline{u}(t)} + \beta(u_T(t)-\overline{u}_T(t))\right) - \alpha\lambda w(t). \tag{3.8}$$

Equation (3.8) will be called the spike-based learning rule, as the postsynaptic spike train $Y(t)$ explicitly appears. An accompanying rate-base learning rule can also be derived:

$$\frac{d}{dt}w(t) = \alpha\frac{1}{u_0}\nu(t)\left(-\frac{u(t)-\overline{u}(t)}{\overline{u}(t)} + \beta(u_T(t)-\overline{u}_T(t))\right) - \alpha\lambda w(t). \tag{3.9}$$

3.3 Analytical Results

The learning rules (3.8) and (3.9) are stochastic differential equations for the weights w_j driven by the processes $Y(.)$, $\nu_j(.)$ and $u_T(.)$, of which the last two are assumed to be stationary with the means $\langle\nu_j(t)\rangle = \nu_0$ and $\langle u_T(t)\rangle = u_{T,0}$ respectively. The evolution of the solutions $w(t)$ to (3.8) and (3.9) may be studied via a Master equation for the probability distribution of the weights $p(w,t)$ (see (Risken, 1996)). For small learning rates α, the stationary distribution $p(w)$ sharply peaks[4] at the roots of the drift function $A(w)$ of the corresponding Fokker-Planck equation (the detailed derivation is given in the appendix C). Thus, for $\alpha \ll 1$, the temporal evolution of the learning rules (3.8) and (3.9) may be studied via the deterministic differential equation:

$$\frac{d}{dt}\hat{w} = A(\hat{w}) = \alpha\frac{\nu_0}{u_0}\left(-\frac{1}{z}C\hat{w} + \beta C^T\right) - \alpha\lambda\hat{w} \tag{3.10}$$

$$z = \sum_{j=1}^{N} w_j, \tag{3.11}$$

where z is the total weight. The matrix C (with the elements C_{ij}) is the covariance matrix of the input and the vector C^T quantifies the covariance between the activities ν_j and the trace u_T:

$$C_{ij} = \frac{1}{\nu_0^2}\langle\nu_i(t),\nu_j(t)\rangle$$
$$C_j^T = \frac{1}{\nu_0}\langle\nu_j(t),u_T(t)\rangle.$$

appendix C.

[4] A root w^* of $A(w)$ may also correspond to a minimum of $p(w)$ at w^*, if the critical point w^* of (3.10) is unstable. Thus, we are only interested in stable critical points of (3.10).

Now the critical points w^* of dynamics of (3.10) are investigated. These critical points, if asymptotically stable, determine the peaks of the stationary distribution $p(w)$ of the weights w; we therefore expect the solutions of the stochastic equations to fluctuate around these fixed points w^*. If β and λ are much larger than one, the term containing the matrix C can be neglected and equation (3.10) has a unique stable fixed point w^*:

$$w^* = \frac{\nu_0 \beta}{u_0 \lambda} C^T.$$

Under this assumption the maximal mutual information between the target signal $Y_T(t)$ and the output of the neuron $Y(t)$ is obtained by a weight vector $w = w^*$ that is parallel to the covariance vector C^T.

In general, the critical points w^* of (3.10) can be determined using the following fact: Since the covariance matrix C of the input X is symmetric, every element in $\mathbb{R}^N$ can be expressed as a linear combination of eigenvectors b^i of C with eigenvalues λ_i, thus $C^T = \sum_i \pi_i b^i$ with coefficients π_i. The critical point w^* can be written as the linear combination $w^* = \sum_i \alpha_i b^i$, with the coefficients α_i:

$$\alpha_i = \frac{\beta \pi_i z^*}{\lambda_i + \lambda \frac{u_0}{\nu_0} z^*}. \tag{3.12}$$

The total weight $z^* = \sum_j w_j^*$ is the solution of the algebraic equation:

$$\sum_{i=1}^{N} \frac{\beta \pi_i \overline{b}^i}{\lambda_i + \lambda \frac{u_0}{\nu_0} z^*} = 1, \tag{3.13}$$

where $\overline{b}^i = \sum_j b_j^i$. It can be shown (see appendix C for proof) that there is a unique critical point w^* satisfying (3.12) and (3.13); further, this critical point is asymptotically stable. Thus, a stationary unimodal[5] distribution $p(w)$ of the weights w is predicted, which is centered around the value w^*.

3.4 A Concrete Example for IB Optimization

A special scenario of interest, that often appears in the literature (see for example (Klampfl et al., 2007), (Gütig, Aharonov, Rotter, & Sompolinsky, 2003) and (Meffin, Besson, Burkitt, & Grayden, 2006)), is the following: The synapses, and subsequently the input spike trains, form M different subgroups $G_l,\ l \in \{1, \ldots, M\}$ of the same size $N/M \in \mathbb{N}$. The spike trains X_j and X_k, $j \neq k$, are statistically independent if they belong to different subgroups; within a subgroup there is a homogeneous covariance term $C_{jk} = c_l,\ j \neq k$ for $j, k \in G_l$, which can be due either to spike-spike correlations or correlations in rate modulations. The covariance between the target signal and the spike trains X_j is homogeneous among a subgroup. Thus C^T is of the form $C^T = \sum_{l=1}^{M} \pi_l e^l$, with coefficients π_l. e^l is defined as the vector, whose component $e_j^l,\ j \in \{1, \ldots, N\}$ is

[5] Note that $p(w)$ denotes the distribution of the weight vector, not the distribution of a single weight $p(w_j)$.

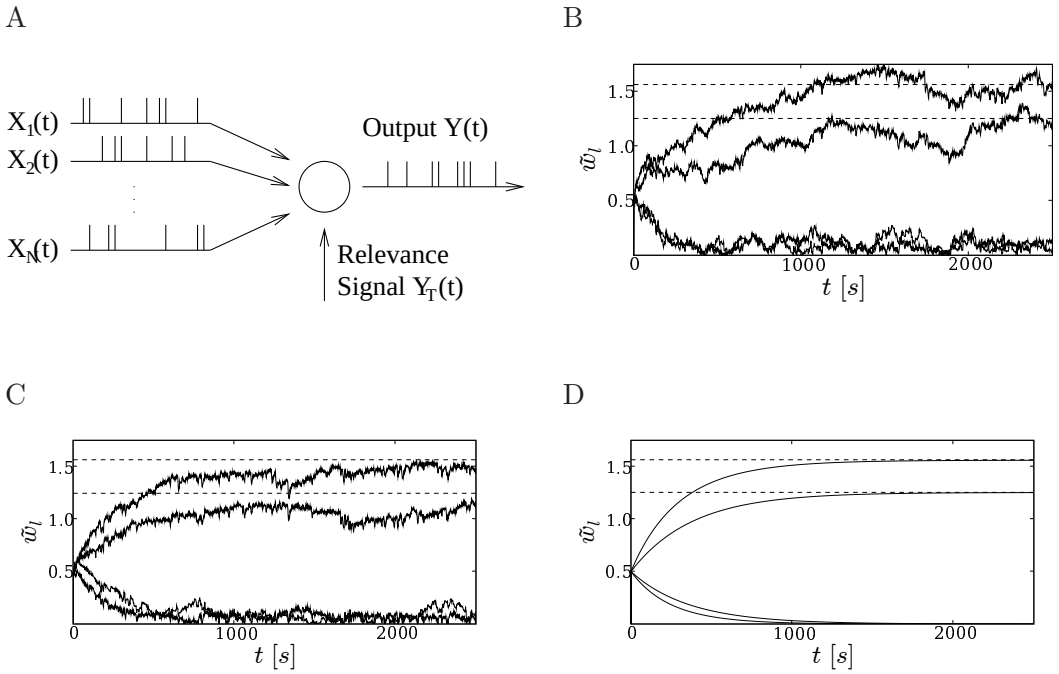


Figure 3.1: A: The basic setup for the Information Bottleneck optimization. B-D: Numerical and analytical results for the IB optimization task described in section 3.4. The temporal evolution of the average weights $\tilde{w}_l = 1/M \sum_{j \in G_l} w_j$ of the four different synaptic subgroups G_l is shown. B: The performance of the spike-based rule (3.8). The highest trajectory corresponds to $\tilde{w}_1$; it stays close to its analytical predicted fixed point value obtained from (3.12) and (3.13), which is visualized by the upper dashed line. The trajectory just below belongs to $\tilde{w}_3$, for which the fixed point value is also plotted as dashed line. The other two trajectories $\tilde{w}_2$ and $\tilde{w}_4$ decay and eventually fluctuate above the predicted value of zero. C: The performance of the rate-based rule (3.9); results are analogous to the ones of the spike-based rule. D: Simulation of the deterministic equation (3.10).

one if $j \in G_l$ and zero otherwise. In this setup, the e^l are eigenvectors of C and the results from section 3.3 can immediately be applied: The unique stable critical point w^* is determined via (3.12) with the total weight $z^* = \sum_j w_j^*$ given by (3.13), using $\overline{b}^i = \overline{e}^i = M$.

As a numerical example, we consider in Fig. 3.1 a modification of the IB task presented in Fig. 2 of (Klampfl et al., 2007). The $N = 100$ synapses form $M = 4$ subgroups $G_l = \{25(l-1)+1, \ldots, 25l\}$, $l \in \{1, \ldots, 4\}$. Synapses in G_1 receive Poisson spike trains of constant rate $\nu_0 = 20\,\mathrm{Hz}$, which are mutually spike-spike correlated with a correlation-coefficient[6] of 0.5. The same holds for the spike trains of G_2. Spike trains for G_3 and G_4 are uncorrelated Poisson trains with a common rate modulation, which is equal to low pass filtered white noise (cut-off frequency 5 Hz) with mean ν_0 and standard deviation (SD) $\sigma = \nu_0/2$. The rate modulations for G_3 and G_4 are however independent (though identically distributed). Two spike trains for different synapse subgroups are statistically independent. The target signal Y_T was chosen to be the sum of two Poisson trains. The first is of constant rate ν_0 and has spike-spike correlations with G_1 of coefficient 0.5; the second is a Poisson spike train with the same rate modulation as the spike trains of G_3 superimposed by additional white noise of SD 2 Hz. Furthermore, the target signal was turned off during random intervals[7]. The resulting evolution of the weights is shown in Fig. 3.1, illustrating the performance of the spike-based rule (3.8) as well as of the rate-based rule (3.9). As expected, the weights of G_1 and G_3 are potentiated as Y_T has mutual information with the corresponding part of the input. The synapses of G_2 and G_4 are depressed. The analytical result for the stable fixed point w^* obtained from (3.12) and (3.13) is shown as dashed lines and is in good agreement with the numerical results. Furthermore the trajectory of the solution $\hat{w}(t)$ to the deterministic equation (3.10) is plotted.

This IB task was slightly changed from the one presented in (Klampfl et al., 2007), because for the setting used here, the covariance matrix C and C^T can be calculated analytically. The simulation results for the original setting in (Klampfl et al., 2007) can also be reproduced with the simpler rules (3.8) and (3.9) (not shown).

3.5 Relevance-Modulated PCA with Spiking Neurons

The presented learning rules (3.8) and (3.9) exhibit a close relation to Principal Component Analysis (PCA). A learning rule which enables the linear Poisson neuron to extract principal components from the input $X(.)$ can be derived by maximizing the following objective function:

$$L_{\mathrm{PCA}} = -L_{\mathrm{IB}} - \lambda \sum_{j=1}^{N} w_j^2 = +I(X,Y) - \beta I(Y_T, Y) - \lambda \sum_{j=1}^{N} w_j^2, \tag{3.14}$$

[6] Spike-spike correlated Poisson spike trains were generated according to the method outlined in (Gütig et al., 2003).

[7] These intervals were Poisson distributed (i. e. there is a constant probability density in time for turning-off Y_T) with a mean rate of 10^{-2} Hz and were of random duration, which was normal distributed with mean 5 s and SD 1 s.

which just differs from (3.3) by a change of sign in front of L_{IB}. The resulting learning rule is in close analogy to (3.8):

$$\frac{d}{dt}w(t) = \alpha\frac{Y(t)}{u(t)}\nu(t)\left(\frac{u(t)-\overline{u}(t)}{\overline{u}(t)} - \beta(u_T(t)-\overline{u}_T(t))\right) - \alpha\lambda w(t). \tag{3.15}$$

The corresponding rate-based version can also be derived. Without the trace $u_T(.)$ of the target signal, it can be seen that the solution $\hat{w}(t)$ of deterministic equation corresponding to (3.15) (which is of the same form as (3.10) with the obvious sign changes) converges to an eigenvector of the covariance matrix C. Thus, for $u_T(.) = 0$ we expect the learning rule (3.15) to perform PCA for small learning rates α. The rule (3.15) without the relevance signal is comparable to other PCA rules, e. g. the covariance rule (see (Sejnowski & Tesauro, 1989)) for non-spiking neurons.

The side information given by the relevance signal $Y_T(.)$ can be used to extract specific principal components from the input, thus we call this paradigm relevance-modulated PCA. Before we consider a concrete example for relevance-modulated PCA, we want to point out a further application of the learning rule (3.15).

The target signal Y_T can also be used to extract different components from the input with different neurons (see figure 3.2). Consider m neurons receiving the same input X. These neurons have the outputs $Y_1(.), \ldots, Y_m(t)$, target signals $Y_T^1(.), \ldots, Y_T^m(t)$ and weight vectors $w^1(t), \ldots, w^m(t)$, the latter evolving according to (3.15). In order to prevent all weight vectors from converging towards the same eigenvector of C (the principal component), the target signal Y_T^i for neuron i is chosen to be the sum of all output spike trains except Y_i:

$$Y_T^i(t) = \sum_{j=1,\ j\neq i}^{N} Y_j(t). \tag{3.16}$$

If one weight vector $w^i(t)$ is already close to the eigenvector e^k of C, than by means of (3.16), the basins of attraction of e^k for the other weight vectors $w^j(t),\ j \neq i$ are reduced (or even vanish, depending on the value of β). It is therefore less likely (or impossible) that they also converge to e^k. In practice, this setup is sufficiently robust, if only a small number (≤ 4) of different components is to be extracted and if the differences between the eigenvalues λ_i of these principal components are not too big. For the PCA learning rule, the time constant τ_0 of the kernel κ_1 (see (3.7)) had to be chosen smaller than for the IB tasks in order to obtain good performance; we used $\tau_0 = 10\,\mathrm{ms}$ in simulations. This is in the range of time constants for IPSPs. Hence, the signals Y_T^i could probably be implemented via lateral inhibition.

The learning rule considered in (Klampfl et al., 2009) displayed a close relation to Independent Component Analysis (ICA). Because of the linear neuron model used here and the linearization of further terms in the derivation, the resulting learning rule (3.15) performs PCA instead of ICA.

The results of a numerical example are shown in figure 3.2. The $m = 3$ for the regular PCA experiment neurons receive the same input X and their weights change according to (3.15). The weights and input spike trains are grouped into four subgroups

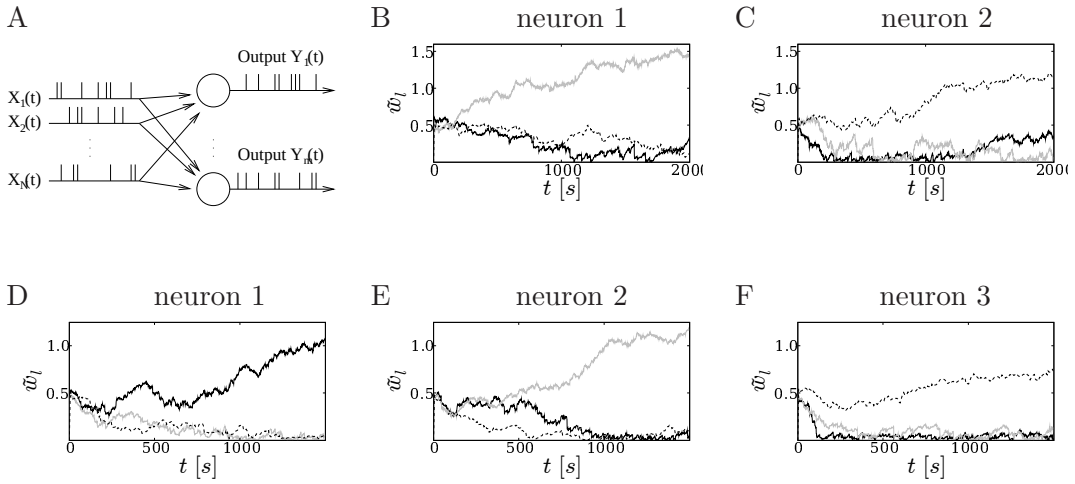


Figure 3.2: A: The basic setup for the PCA task: The m different neurons receive the same input X and are expected to extract different principal components of it. B-F: The temporal evolution of the average subgroup weights $\tilde{w}_l = 1/25 \sum_{j \in G_l} w_j$ for the groups G_1 (black solid line), G_2 (light gray solid line) and G_3 (dotted line). B-C: Results for the relevance-modulated PCA task: neuron 1 (fig. B) specializes on G_2 and neuron 2 (fig. C) on subgroup G_3. D-F: Results for the regular PCA task: neuron 1 (fig. D) specialize on G_1, neuron 2 (fig. E) on G_2 and neuron 3 (fig. F) on G_3 .

$G_1, \ldots, G_4$, as for the IB optimization discussed in section 3.4. The only difference is that all groups (except for G_4) receive spike-spike correlated Poisson spike trains with a correlation coefficient for the groups G_1, G_2, G_3 of 0.5, 0.45, 0.4 respectively. Group G_4 receives uncorrelated Poisson spike trains. As can be seen in figure 3.2 **D** to **F**, the different neurons specialize on different principal components corresponding to potentiated synaptic subgroups G_1, G_2 and G_3 respectively. Without the relevance signals $Y_T^i(.)$, all neurons tend to specialize on the principal component corresponding to G_1 (not shown).

As a concrete example for relevance-modulated PCA, we consider the above setup with slight modifications: Now we want $m = 2$ neurons to extract the components G_2 and G_3 from the input X, and not the principal component G_1. This is archived with an additional relevance signal Y_T^0, which is the same for both neurons and has spike-spike correlations with G_2 and G_3 of 0.45 and 0.4. We add the term $\gamma I(Y, Y_T^0)$ to the objective function (3.14), where γ is a positive trade-off factor. The resulting learning rule has exactly the same structure as (3.15), with an additional term due to Y_T^0. The numerical results are presented in Fig. 3.2 **B** and **C**, showing that it is possible in this setup to explicitly select the principle components that are extracted (or not extracted) by the neurons.

3.6 Discussion

We have introduced and analyzed a simple and perspicuous rule that enables spiking neurons to perform IB optimization in an online manner. Our simulations show that this rule works as well as the substantially more complex learning rule that had previously been proposed in (Klampfl et al., 2009). It also performs well for more realistic neuron models as indicated in the appendix C. We have shown that the convergence properties of our simplified IB rule can be analyzed with the help of the Fokker-Planck equation (alternatively one can also use the theoretical framework described in A.2 in (Intrator & Cooper, 1992) for its analysis). The investigation of the weight vectors to which this rule converges reveals interesting relationships to PCA. Apparently, very little is known about learning rules that enable neurons to perform PCA (a discussion of a special case is given in chapter 11.2.4 of (Gerstner & Kistler, 2002)). We have demonstrated both analytically and through simulations that a slight variation of our new learning rule performs PCA. Our derivation of this rule within the IB framework opens the door to new variations of PCA where preferentially those components are extracted from a high dimensional input stream that are –or are not– related to some external relevance variable.

We expect that a further investigation of such methods will shed light on the unknown principles of unsupervised and semi-supervised learning that might shape and constantly retune the output of lower cortical areas to intermediate and higher cortical areas. The learning rule that we have proposed might in principle be able to extract from high-dimensional sensory input streams X those components that are related to other sensory modalities or to internal expectations and goals.

Quantitative biological data on the precise way in which relevance signals Y_T (such as for example dopamin) might reach neurons in the cortex and modulate their synaptic plasticity are still missing. But it is fair to assume that these signals reach the synapse in a low-pass filtered form of the type u_T that we have assumed for our learning rules. From that perspective one can view the learning rules that we have derived (in contrast to the rules proposed in (Klampfl et al., 2009)) as local learning rules.

3.7 Acknowledgments

This chapter is based on the paper *Simplified rules and theoretical analysis for information bottleneck optimization and PCA with spiking neurons*, which was written by Lars Büsing (LB) and Wolfgang Maass (WM). The model was developped by LB. The simulations were designed by LB and WM and conducted by LB. LB and WM wrote the manuscript.

Chapter 4

Extended Spiking Information Bottleneck

Contents

Neurons receive thousands of presynaptic input spike trains while emitting a single output spike train. This drastic dimensionality reduction suggests to consider a neuron as a bottleneck for information transmission. Extending recent results, we propose a simple learning rule for the weights of spiking neurons derived from the Information Bottleneck (IB) framework that minimizes the loss of *relevant* information transmitted in the output spike train. In the IB framework relevance of information is defined with respect to contextual information, the latter entering the proposed learning rule as a "third" factor besides pre- and postsynaptic activities. This renders the theoretically motivated learning rule a plausible model for experimentally observed synaptic plasticity phenomena involving three factors. Furthermore, we show that the proposed IB learning rule allows spiking neurons to learn a "predictive code", i.e. to extract those parts of their input that are predictive for future input.

Information theory is a powerful theoretical framework with numerous important applications, also in the context of neuroscience such as the analysis of experimental data. Furthermore, information theory has also provided rigorous principles for learning in abstract and more biological realistic models of neural networks. Especially the learning objective of maximizing information transmission of single neurons and neural networks, a principle often termed InfoMax, has been intensively studied in (Linsker, 1989; Bell & Sejnowski, 1995; Chechik, 2003; Toyoizumi, Pfister, Aihara, & Gerstner, 2005; Parra, Beck, & Bell, 2009). This learning principle has been shown to be a possible framework for Independent Component Analysis and furthermore it could successfully explain aspects of synaptic plasticity experimentally observed in neural tissue. However one limitation of this learning objective for gaining a principled understanding of computational processes in neural systems is the fact, that the goal of numerous types of computations is not a maximization of information transmission (e.g. from sensory

input neurons to areas in the brain where decision are made). Rather, a characteristic feature of generic computations (e.g. clustering and classification of data, or sorting a list of elements according to some relation) is that they remove some of the information contained in the input. Similarly, generic learning processes require the removal of some of the information originally available in order to achieve generalization capability.

(Tishby et al., 1999) created a new information theoretic framework, the Information Bottleneck (IB) framework, that focuses on transmitting the maximal amount of *relevant* information. This approach takes a step towards making computational and learning processes more amenable to an information theoretic analysis. We examine in this article the question whether the IB framework can foster the understanding of organizational principles behind experimentally verified synaptic plasticity mechanisms which involve a "third factor" (Sjöström & Häusser, 2006; Hee et al., 2007). These are plasticity effects where the amplitude the of synaptic weight change does not only depend on the firing activity of the pre- and postsynaptic neuron, but in addition on a third signal that is transmitted e.g. in form of neuromodulators or synaptic inputs from other neurons. Such third signals are known, for example, to modulate the amplitude of the backpropagating action potential, and thereby to critically influence the changes of synaptic weights elicited by spike timing-dependent plasticity (STDP). Furthermore, we examine in this article the question whether one can derive from IB principles a rule for synaptic plasticity that establishes a generic computation in neural circuits: the extraction of temporally stable ("slow") sensory stimuli (see e.g. (Wiskott & Sejnowski, 2002)).

The extraction of relevant features and the neglect of irrelevant information from given data is a common problem in machine learning and it is also widely believed to be an essential step of neural processing of sensory input streams. However, which information contained in the input data is to be considered relevant is of course highly dependent on the context. In the seminal paper (Tishby et al., 1999) an information theoretic definition of relevance wrt. to a given context was proposed, and furthermore a batch algorithm for data compression minimizing the loss of relevant information was presented. This framework, the Information Bottleneck method, is aimed at constructing a simple, compressed representation Y (relevant features, the Information Bottleneck) of the given input data X which preserves high mutual information with a relevance (or target) signal R which provides the contextual or side information. In the IB framework the amount of relevant information contained in a random variable is explicitly defined as the mutual information of this variable with the relevance signal R. Multiple algorithms rooting in the IB framework have been fruitfully applied to typical machine learning applications such as document clustering, document classification, image classification and feature extraction for speech recognition (see (Harremoes & Tishby, 2007) and references therein).

Recently, it has been conjectured that the IB framework might constitute one of the optimization principles underlying early neural processing of sensory input data in some organisms. In (Bialek, Steveninck, & Tishby, 2006) it is argued that biological agents maintain an internal representation of the external world that exclusively contains information important for their survival capabilities. More precisely it is hypothesized

that only those parts of the sensory input X should be internally represented in some model Y that are predictive for the future state of the agent's environment as only this information is relevant for the agent's future actions which in turn increase its fitness. This learning paradigm was formalized as an IB optimization with the relevance signal R defined as the future sensory stimuli. As an IB optimal internal representation Y, called a "predictive code", apparently depends strongly on the statistics of the environment and as many organisms exhibit a remarkable ability to adapt to different environmental configurations it is tempting to conjecture that the internal representation Y is (at least partially) learned during the lifetime of the agent. However in the studies mentioned above learning rules for developing this kind of internal representation in a biologically realistic setting, where the standard batch IB algorithms are implausible, are missing.

An attempt to fill this apparent gap has been made in (Klampfl et al., 2009) on the level of single spiking neuron models. In this chapter a single neuron is considered as an information bottleneck, as it maps its high dimensional input X to its one dimensional output spike train Y. Based on this interpretation, an online update rule has been proposed which adjusts the synaptic weights such that the neuron's output Y contains the maximal amount of relevant information wrt. to a given relevance signal R, which was also modeled as a spike train. This learning rule has been shown to reliably solve numerous concrete IB optimization problems in a neural context. However the proposed learning rule, which was derived by stochastic gradient ascent on the IB objective function (essentially the amount of transmitted relevant information), has several drawbacks. The gradient of the transmitted relevant information (which determines the learning rule) was estimated using the correlation of the bottleneck neuron output Y and the relevance signal R within each single time step. This limits the "complexity" of IB problems that the neuron is able to solve, e.g. this estimation cannot capture long delays between the input X and the relevance signal R nor can it capture the impact of higher order moments between the input X and the relevance variable R in the case of linear bottleneck neurons. Furthermore, the learning rule of (Klampfl et al., 2009) is complicated, making it difficult to understand the learning dynamics. In addition it contains non-local variables which reduces its biological plausibility.

The goal of this chapter is to develop a simpler and more transparent approximate IB learning rule for spiking neurons. This new IB learning rule is based on a different estimation of the gradient of the relevant information contained in the neural output. The estimation is of parametric nature, and it requires a given preprocessing of the relevance signal R. The main assumption of this chapter is hence that the "bottleneck neuron" has access to a rich preprocessing of the relevance signal R. This preprocessing can be considered as a "third" factor, besides the presynaptic input X and the output Y, which modulates synaptic plasticity in order to implement an IB optimal coding of the inputs.

The outline of the chapter is the following. In section 4.1 the IB framework is briefly revisited, the underlying spiking neuron model is defined and the objective function for IB optimization for spiking neurons is introduced. We present the general IB learning rule for spiking neurons in section 4.2 and discuss a concrete, simple example IB task. Further, in this section we propose an implementation of the relevance signal prepro-

cessing using a generic recurrent neural network. In section 4.3 the proposed IB learning rule is used to model the learning of a predictive code. A detailed comparison to related work is presented in section 4.4. Furthermore, experimental results on synaptic plasticity with three factors that point out a possible implementation of the learning rule proposed in this chapter are discussed.

4.1 Neuron Model and Objective Function

In this section, the neuron model and the objective function for IB optimization with this model are defined. The model is formulated in discrete time of step size Δt. To introduce a biologically plausible time scale we assume that a single time step corresponds to one millisecond, i.e. $\Delta t = 1\,\text{ms}$. The value of a time-varying function f at time step t will be denoted as f^t. Further, the standard euclidean dot product of two vectors $\boldsymbol{d} = (d_1, \ldots, d_N)$ and $\boldsymbol{e} = (e_1, \ldots, e_N)$ is written as $\boldsymbol{d} \cdot \boldsymbol{e} := \sum_{i=1}^{N} d_i e_i$. We start this section by briefly revisiting the Information Bottleneck method.

4.1.1 The Information Bottleneck Method

The Information Bottleneck (IB) method, that was originally introduced in (Tishby et al., 1999), is a data compression technique which, in its simplest version, focuses on the following setup. Consider two random variables (RVs) a and b with a known joint distribution $p(a, b)$. The goal of the IB method is to construct a RV $\tilde{a}$ which is a compact, simple representation of a via a stochastic mapping defined by the conditional probability $p(\tilde{a}|a)$ such that $\tilde{a}$ still is informative about b. The RV b will be called the *relevance* or *target* signal in the remainder of this chapter. This intuitive data compression task was formalized in (Tishby et al., 1999) as a maximization problem of the objective function L_{IB}:

$$L_{\text{IB}} = I(\tilde{a}, b) - \gamma I(\tilde{a}, a).$$

Here $I(.,.)$ denotes the mutual information between the two arguments. The first term $I(\tilde{a}, b)$ of L_{IB} measures how informative the compressed representation $\tilde{a}$ is about b. The second term $I(\tilde{a}, a)$ with the Lagrange multiplier $\gamma > 0$ penalizes complex representations $\tilde{a}$ and can be regarded as an information theoretic regularization term [1]. The IB method consists in finding a conditional probability distribution $p(\tilde{a}|a)$ that maximizes L_{IB} under the condition that $\tilde{a}$, a and b form the Markov chain $b \to a \to \tilde{a}$.[2] The parameter $\gamma \in [0, 1]$ determines the degree of compression via the trade-off between the relevant information that $\tilde{a}$ carries about b and the complexity of $\tilde{a}$. For $\gamma = 0$ the representation $\tilde{a}$ is uncompressed and all relevant information is preserved, i.e. L_{IB} is maximal e.g. for the identity mapping and $I(\tilde{a}, b) = I(a, b)$. On the other extreme, for $\gamma = 1$ the variable $\tilde{a}$ is maximally compressed and always assumes a single value resulting in $I(\tilde{a}, a) = 0$ and $I(\tilde{a}, b) = 0$.

[1] The IB objective function in (Tishby et al., 1999) was originally introduced with the opposite sign and it was parametrized in terms of $\beta := \gamma^{-1}$.

[2] This condition is equivalent to requiring $\tilde{a}$ to be independent from b given a.

An application in machine learning, which illustrates the merits of the IB method, is feature selection for document classification as presented in (Slonim & Tishby, 2001). In this setup, the uncompressed input a corresponds to words, which occur in the documents, and the relevance variable b is chosen to be the class label, i.e. the document category, e.g. "sports" or "politics"; the joint distribution of words and document categories $p(a, b)$ is assumed to be known for a given training set. Via the IB method it is possible to obtain a mapping $p(\tilde{a}|a)$ yielding a simple representation $\tilde{a}$ (word clusters instead of single words) which still carries most of the relevant information about the document class. These low-dimensional word-clusters can then be conveniently used as features for document classification of test data.

4.1.2 Neuron Model

We consider a simple, stochastic neuron model similar to the ones used in (Toyoizumi et al., 2005) and (Klampfl et al., 2009), however without taking a refractory mechanism into account. The neuron has N synapses with weights $\boldsymbol{w} = (w_1, \ldots, w_N)$, which we require to be non-negative. It is driven by the input $X = (X_1, \ldots, X_N)$, consisting of N spike trains $X_j = (\ldots, x_j^{-1}, x_j^0, x_j^1, \ldots)$ formalized as left and right infinite sequences. We define $x_j^t = 1$ if there is a presynaptic spike at synapse j at time step t and $x_j^t = 0$ otherwise. The spikes at synapse j from time step l up to t ($l < t$) are written as $X_j^{l,t} = (x_j^l, x_j^{l+1}, \ldots, x_j^t)$, further the input history up to time step $t = 0$ of synapse j is denoted as $X_j^{-\infty} := (\ldots, x_j^{-1}, x_j^0) = X_j^{-\infty,0}$ and $X^{-\infty} = (X_1^{-\infty}, \ldots, X_N^{-\infty})$. The membrane potential u^t of the neuron at time t is given by the weighted sum of the synaptic activities $\boldsymbol{\nu}^t = (\nu_1^t, \ldots, \nu_N^t)$:

$$\begin{aligned} u^t &= \boldsymbol{w} \cdot \boldsymbol{\nu}^t = \sum_{j=1}^{N} w_j \nu_j^t \\ \nu_j^t &= (\epsilon * X_j)^t = \sum_{l=-\infty}^{\infty} \epsilon^l x_j^{t-l}. \end{aligned} \tag{4.1}$$

The kernel ϵ models the EPSP of a single spike and $*$ denotes the discrete time convolution. Given the input X, the postsynaptic neuron spikes at time step t with the probability $p(y^t = 1|X^{-\infty,t})$, which is a function of the membrane potential:

$$p(y^t = 1|X^{-\infty,t}) = g(u^t) = g^t.$$

The function g is called the activation function. Its image is assumed to be in $[0, 1]$ and further it is assumed to be continuously differentiable with a derivative $g'(u^t) =: g'^t$. The postsynaptic spike train is denoted as $Y = (\ldots, y^{-1}, y^0, y^1, \ldots)$ with $y^t = 1$ if an output spike occurs at time step t, and 0 otherwise. In simulations the EPSP kernel ϵ was chosen to be a non-anticipating, decaying exponential with a time constant of 10 time steps and the activation function $g(u^t) = \sigma(u^t - u_0)$ was chosen as the logistic function $\sigma(x) := (1 + \exp(-x))^{-1}$ with an offset $u_0 = -2$.

Furthermore, according to the IB framework, another external signal besides the in-

put X is given, namely the *relevance* or *target* signal R. We consider situations where R is given by a stochastic process denoted by the sequence $R = (\ldots, R^{-1}, R^0, R^1, \ldots) \in \mathbb{R}^{\mathbb{Z}}$. It is not restricted to spike trains, it may be given by a more general real valued sequence. It is straight forward to extend the results presented below to multi-dimensional relevance variables. We assume that R does not directly influence the activity of the neuron, but that it only takes part in the process of the synaptic plasticity in order to ensure that $R \to X \to Y$ is a Markov chain as required by the IB framework. For the sake of simplicity we assume that the processes X and R are stationary.

4.1.3 Applying the IB framework to Spiking Neurons

Following the approach taken by (Klampfl et al., 2009), we apply the IB framework to a single neuron as illustrated in Fig. 4.1. At any given time step t, without loss of generality we may assume $t = 0$, the neuron under consideration, which we call the bottleneck neuron from now on, maps its input history $X^{-\infty}$ to an output $y^0 \in \{0, 1\}$. Hence the neuron can be regarded as an information bottleneck, which compresses its high-dimensional input history $X^{-\infty}$ (corresponding to a in the notation introduced in section 4.1.1) to its one-dimensional binary output y^0 (corresponding to $\tilde{a}$). This mapping is parametrized by the weight vector $\boldsymbol{w}$ for which we want to find the configuration giving rise to the output of the neuron that is maximally informative about the relevance signal R (corresponding to b in section 4.1.1). There are multiple possible ways of formalizing this setup in the IB framework, more precisely of choosing the IB objective function.

The Choice of the IB Objective Function for Spiking Neurons

We define the amount of relevant information that is transmitted by the neuron per time step as the mutual information $I(y^0, R)$ between the current output y^0 and the whole relevance signal R. Hence, following the IB framework we are looking for the synaptic weight $\boldsymbol{w}$ that maximizes the IB objective function L_{IB} with a regularization term L_{reg}:

$$\begin{aligned} L_{\text{IB}} &= I(y^0, R) - \gamma L_{\text{reg}} \\ &= \left\langle \log\left(\frac{p(y^0|R)}{p(y^0)}\right)\right\rangle - \gamma L_{\text{reg}}, \end{aligned} \tag{4.2}$$

where the brackets $\langle . \rangle$ denote the expected value over the input spike trains X, the output spike train Y and the relevance signal R. Further $p(y^0)$ and $p(y^0|R)$ denote the unconditioned spiking probability and the spiking probability conditioned on the relevance signal respectively.

Our definition of the relevant information as $I(y^0, R)$ can be interpreted as the limit of the mutual information $I(y^0, R^{-T,T})$ between y^0 and the relevance signal $R^{-T,T}$ in a time window of length $2T + 1$ for $T \to \infty$ (i.e. $I(y^0, R) = \lim_{T\to\infty} I(y^0, R^{-T,T})$). This choice eliminates "cut-off" artifacts like the following: If the relevant information contained in the input X arrived at the bottleneck neuron with a delay of $T+1$ relative to the relevance signal R, the objective function $I(y^0, R^{-T,T})$ would be insensitive to this statistical relation. One might reckon that the choice to maximize $I(y^0, R)$ introduces

anticipatory effects, e.g. that for adapting its weights $\boldsymbol{w}$ the bottleneck neuron would need information that will only be available in the future. Such effects will however not show up in our approach as it is explicitly designed to only take into account information for the IB optimization that is currently available to the neuron, as outlined in the next section.

Alternative definitions of the relevant information are also possible of course. It might be argued that e.g. the mutual information $I(y^{-T,T}, R^{-T,T})$ between the neuron output and the relevance signal in some time window is a more natural definition of the relevant information. However, it turned out that the online optimization of $I(y^{-T,T}, R^{-T,T})$ is considerably more difficult than the one of $I(y^0, R)$ due to accounting for relations between multiple outputs spikes of the bottleneck neuron. These technical difficulties motivated the choice of optimizing $I(y^0, R)$.

As stated in section 4.1.1 the regularization L_{reg} in the original IB formulation from (Tishby et al., 1999) was given by the mutual information between the input and the output of the IB mapping, i.e. in this setup $L_{\text{reg}} = I(y^0, X^{-\infty})$. This choice is also possible in the neural context considered here. However, simulation results indicate that for this definition of L_{reg} sensible values of the trade-off parameter γ, i.e. those values of γ that neither "un-regularize" nor "over-regularize" (basically the $\boldsymbol{w} = 0$) the IB optimization, are confined to a small interval and are thus hard to be determined numerically. Therefore we replace the original regularization with a conventional quadratic regularization of the weights $\boldsymbol{w}$:

$$L_{\text{reg}} = \frac{1}{2}\boldsymbol{w}^2.$$

With this choice of L_{reg} it is considerably easier to determine sensible values of γ. Other choices of L_{reg} are also possible such as penalizing deviations from an average target firing rate. It is straight forward to incorporate such a regularization into the objective function presented here.

Online Estimation of the Relevant Information

Eventually, we wish to maximize L_{IB} defined in (4.2) via a stochastic gradient ascent wrt. the weights $\boldsymbol{w}$ yielding an online update rule. However, this maximization requires the explicit knowledge of the conditional distribution $p(y^0|R)$ of the output y^0 given the relevance signal R as can be seen from equation (4.2). Most IB algorithms resolve this issue by estimating the joint distribution of the input data and the relevance signal (here $p(X^{-\infty}, R)$) from the whole batch data set and subsequently evaluating the conditional distribution of the compressed output variable given the relevance signal (here $p(y^0|R)$) using the fact that $R \to X \to Y$ is a Markov chain. However, this approach only seems plausible in an offline, batch IB optimization task where the whole data set is available at all times. In the neural setup considered here, we do not want to assume that the neuron has all information about the joint distribution of $X^{-\infty}$ and R, rather it should estimate $p(y^0|R)$ and the relevant information $I(y^0, R)$ online. As this can be arbitrarily difficult (depending on the "complexity" of $p(X^{-\infty}, R)$), we can only hope to solve the neural IB optimization task approximately under some simplifying assumptions.

A possible strategy that addresses the problem outlined above is the following. As its output y^0 is binary, the neuron only has to estimate $p(y^0 = 1|R)$ in order to determine an approximation of $I(y^0, R)$ (and its gradient). We therefore assume the neuron has access to a parametric estimation $F^t \approx p(y^t = 1|R)$ with r parameters $\boldsymbol{q} = (q_1, \ldots, q_r)$. For the sake of simplicity we restrict ourselves to the case where F^t is of the form $F^t = \sigma(\boldsymbol{q} \cdot \boldsymbol{h}^t)$ with σ denoting the logistic function that ensures $F^t \in [0, 1]$. The quantities $h = (h_1, \ldots, h_r)$ are r given filters[3] operating on the relevance sequence R and $\boldsymbol{h}^t = (h_1^t, \ldots, h_r^t)$ denotes their values at time step t. These filters h are a preprocessing of the relevance signal R that is currently available to the neuron. In a neural system it might be implemented by some neural circuitry that carries out transformations of the sequence R. In simulations, h can be modeled e.g. by moving averages or Volterra series of the relevance signal R, concrete examples for such a preprocessing as well as for a preprocessing with a simulated neural circuitry are given below. The concrete choice of the form of the estimator F^t, a linear model in the parameters $\boldsymbol{q}$ followed by the logistic function, results in a simple online learning rule (due to similarities with logistic regression). Other forms of F^t which also allow simple gradient ascent could potentially be worth studying.

Based on the estimator F^t (see Fig. 4.1 for a visualization of this approach), we propose the following objective function L to be maximized wrt. $\boldsymbol{w}$ and $\boldsymbol{q}$:

$$L \;=\; L_F - \gamma L_{\mathrm{reg}} = \left\langle \log\left(\frac{F(y^0, R)}{p(y^0)}\right)\right\rangle - \frac{\gamma}{2}\boldsymbol{w}^2, \tag{4.3}$$

where $F(y^0, R) = (F^0)^{y^0}(1 - F^0)^{1-y^0} \approx p(y^0|R)$. The term γL_{reg} represents the regularization term which remains unchanged from the objective function L_{IB} given in (4.2). The term L_F is the approximation of the relevant information $L_F \approx I(y^0, R)$ based on F^0 and it can easily be shown that the following relation holds:

$$\begin{aligned} L_F = \left\langle \log\left(\frac{F(y^0, R)}{p(y^0)}\right)\right\rangle \;&=\; I(y^0, R) - \left\langle D_{\mathrm{KL}}(p(y^0|R)\|F(y^0, R))\right\rangle \qquad (4.4)\\ \text{with} \qquad D_{\mathrm{KL}}(p(y^0|R)\|F(y^0, R)) \;&=\; \sum_{y^0=0}^{1} p(y^0|R)\log\left(\frac{p(y^0|R)}{F(y^0, R)}\right), \end{aligned}$$

where $D_{\mathrm{KL}}(P\|Q)$ denotes the Kullback-Leibler divergence between P and Q. It can be seen from (4.4) that optimizing L wrt. $\boldsymbol{q}$ for fixed weights $\boldsymbol{w}$ amounts to minimizing the Kullback-Leibler divergence between the estimation $F(y^0, R)$ and the "true" conditional distribution $p(y^0|R)$. The divergence $\left\langle D_{\mathrm{KL}}(p(y^0|R)\|F(y^0, R))\right\rangle$ assumes its unique minimum at $F(y^0, R) = p(y^0|R)$. On the other hand, maximizing L wrt. $\boldsymbol{w}$ for fixed $\boldsymbol{q}$ (i.e. for fixed F^0) can be interpreted as maximizing an estimation L of the "true" IB objective function L_{IB}. It can easily be shown that this is equivalent to maximizing the difference of the transmitted information $I(y^0, X^{-\infty})$ and the Kullback-Leibler

[3]Precisely we define a filter $h_i : \mathbb{R}^{\mathbb{Z}} \to \mathbb{R}^{\mathbb{Z}}$ as a mapping from left-right infinite sequences to left-right infinite sequences with $R \mapsto h_i[R] = (\ldots, h_i[R]^{-1}, h_i[R]^0, h_i[R]^1, \ldots)$. F^t is precisely defined as $F^t = \sigma(\boldsymbol{q} \cdot \boldsymbol{h}[R]^t)$. For the sake of simplicity in the main text the shorter notation is used.

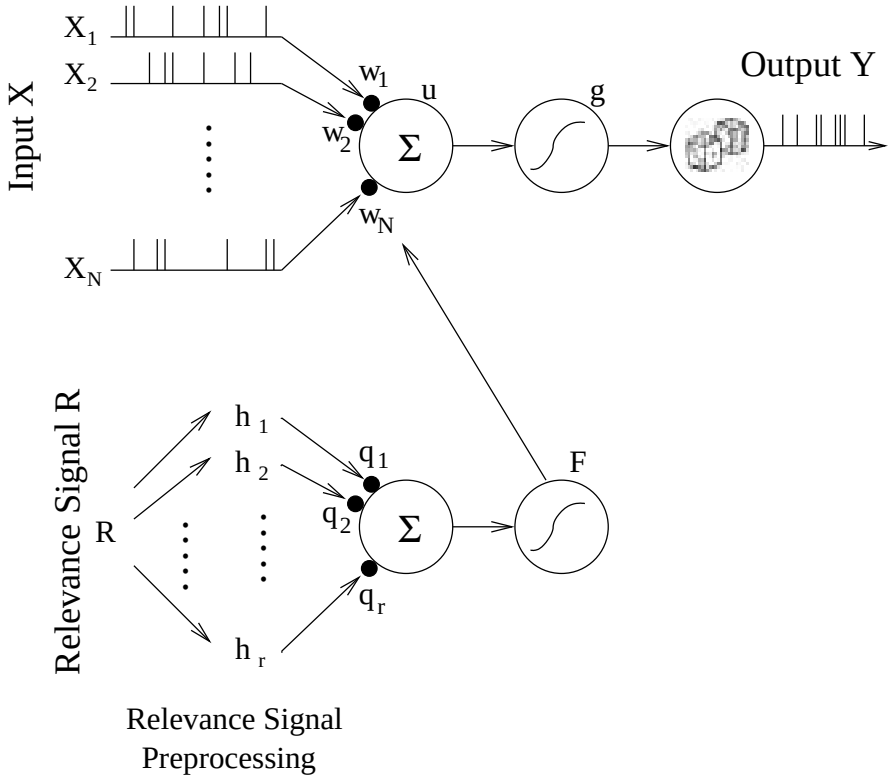


Figure 4.1: The general setup for Information Bottleneck optimization with a spiking neuron. The neuron receives input spike trains X_i for $i = 1, \ldots, N$ and emits the output Y. Furthermore, a second signal, the relevance signal R, is given which allows to introduce the notion of relevance of information. The weights $\boldsymbol{w}$ should be learned such that the relevant information $I(y^0, R)$ that is contained in the neuron output is maximal (under regularization constrains). In order to carry out this optimization in an online manner, an estimation of the gradient of $I(y^0, R)$ wrt. $\boldsymbol{w}$ is required, which is based on the quantity F^0. The latter is a parametrized function (with parameters $\boldsymbol{q}$) of a given preprocessing $h = (h_1, \ldots, h_r)$ of the relevance signal. The parameters $\boldsymbol{q}$ are adapted such that F^0 optimally predicts the neural output y^0 given the relevance signal R.

divergence $\left\langle D_{\mathrm{KL}}(p(y^0|X^{-\infty})\|F(y^0,R))\right\rangle$:

$$L_F \;=\; I(y^0,X^{-\infty}) - \left\langle D_{\mathrm{KL}}(p(y^0|X^{-\infty})\|F(y^0,R))\right\rangle$$

The objective function L further has the following pleasant property:

$$L \leq L_{\mathrm{IB}}$$

This ansatz can hence be understood as maximizing a lower bound L of the "true" IB objective function L_{IB}. In a batch setting, this optimization problem could be solved by an algorithm with two alternating steps that are iterated, reminiscent of the Expectation-Maximization algorithm: In the first step, minimize $\left\langle D_{\mathrm{KL}}(p(y^0|R)\|F(y^0,R))\right\rangle$ wrt. to $\boldsymbol{q}$ for fixed $\boldsymbol{w}$. In the second step minimize $\left\langle D_{\mathrm{KL}}(p(y^0|X^{-\infty})\|F(y^0,R))\right\rangle - I(y^0,X^{-\infty}) + \gamma L_{\mathrm{reg}}$ wrt. to $\boldsymbol{w}$ for fixed $\boldsymbol{q}$. These two steps are iterated until a termination condition is fulfilled.

In the remainder of the chapter we investigate an online optimization scheme for $\boldsymbol{w}$ and $\boldsymbol{q}$ that is obtained by a stochastic gradient ascent on L. For deriving this online learning rule it is advantageous to rewrite L in the following form:

$$L \;=\; \left\langle \log F(y^0,R)\right\rangle + H(y^0) - \frac{\gamma}{2}\boldsymbol{w}^2, \tag{4.5}$$

where $H(y^0)$ denotes the entropy of y^0.

4.2 The IB Learning Rule for Spiking Neurons

4.2.1 The Online Learning Rule

From the objective function L defined in (4.3), an online learning rule for $\boldsymbol{w}$ can be obtained by performing a stochastic gradient ascent with a learning rate η_w:

$$\Delta\boldsymbol{w}^t = \boldsymbol{w}^{t+1} - \boldsymbol{w}^t = \eta_w L_w, \quad \text{with} \quad \langle L_w\rangle = \frac{\partial L}{\partial \boldsymbol{w}},$$

and analogously for the parameters $\boldsymbol{q}$ with a learning rate η_q. As shown in Appendix D.1 this leads to the following equations:

$$\begin{aligned} \Delta\boldsymbol{w}^t &= \eta_w g'^t \boldsymbol{\nu}^t \left(\sigma^{-1}(F^t) - \sigma^{-1}(\left\langle g^t\right\rangle)\right) - \eta_w\gamma\boldsymbol{w} \\ \Delta\boldsymbol{q}^t &= \eta_q \boldsymbol{h}^t (y^t - F^t), \end{aligned} \tag{4.6}$$

where σ^{-1} is the inverse logistic function and $\left\langle g^t\right\rangle$ is the average firing rate of the neuron. Further, g'^t denotes the derivative of the activation function at time step t and $F^t = \sigma(\boldsymbol{q}^t \cdot \boldsymbol{h}^t)$ denotes the estimator of $p(y^t = 1|R)$. For online learning $\left\langle g^t\right\rangle$ is estimated by a running average $\hat{g}^t$ of g^t over an exponential time window of width η_g^{-1}:

$$\hat{g}^{t+1} = (1-\eta_g)\hat{g}^t + \eta_g g^t.$$

Apart from the multiplicative term g'^t, which modulates the amount of weight change

$\Delta \boldsymbol{w}^t$ with the sensitivity (i.e. the derivative) of the activation function, the learning rule (4.6) consists basically of three additive terms. These terms correspond to the gradients of the estimation $\left\langle \log F(y^0, R) \right\rangle \approx -H(y^0|R)$, of the entropy $H(y^0)$ and of the regularization L_{reg} which stem from the three additive terms of L in the form of (4.5). The first term $\boldsymbol{\nu}^t \sigma^{-1}(F^t)$, being proportional to the gradient of $\left\langle \log F(y^0, R) \right\rangle$, increases those weights w_j whose synaptic activity ν_j^t correlates with the estimator F^t (as σ^{-1} is just a monotonic rescaling). Thus those weights are potentiated whose activity can be well predicted by the estimator F^t and hence carry relevant information. The second term, which is proportional to $-\sigma^{-1}(\left\langle g^t \right\rangle) = \log((1 - \left\langle g^t \right\rangle)/\left\langle g^t \right\rangle)$, stemming from the gradient of $H(y^0)$, changes the weights in order to achieve a high entropy of the output y^0 (which is maximal at $\left\langle g^t \right\rangle = 1/2$). This term can be interpreted as a homeostatic control on a long time scale, as the average $\left\langle g^t \right\rangle$ is only slowly changing due to changes of the weights. It pushes the activity of the neuron towards a working regime of optimal information transmission. The last term $-\boldsymbol{w}$ from (4.6) represents the gradient of the regularization $-\boldsymbol{w}^2/2$ and yields a conventional weight decay term.

The update rule for the parameters $\boldsymbol{q}$ is proportional to the difference of the neuron output y^t and F^t. The parameters $\boldsymbol{q}$ assume stationary values if e.g. the estimation F^t fulfills[4] $F^t = \left\langle y^t \right\rangle_{Y|X,R}$.

4.2.2 A Simple Example

In this section, the IB learning rules (4.6) for adapting the weights $\boldsymbol{w}$ and parameters $\boldsymbol{q}$ are applied to a simple IB optimization task. The inputs to the neuron as well as the relevance signal consist of (discrete time) Poisson spike trains. Some of the input spike trains exhibit a statistical dependence on the relevance signal on the level of precise spike times, and hence carry relevant information. Using the learning rule (4.6) the neuron should learn to exclusively potentiate the weights of these input channels and neglect the remaining inputs.

Consider the following setup which is shown in Fig. 4.2. Let the $N = 100$ inputs $X = (X_1, \ldots, X_{100})$ to the bottleneck neuron be grouped into three groups G_1, G_2, G_3 consisting of 25, 25 and 50 neurons respectively. The inputs X_i as well as the relevance signal R are given by spike trains (i.e. binary sequences in $\{0,1\}^{\mathbb{Z}}$) of constant rate[5] $\nu_X = 0.02$ and $\nu_R = 0.06$ (corresponding to 20 Hz and 60 Hz for a time step $\Delta t = 1\,\text{ms}$). Spike trains from different input groups are statistically independent. Furthermore, the inputs are generated such that spike trains from the input groups G_1 and G_2 exhibit a correlation coefficient (CC) with the relevance spike train R of $c_1 = 0.1$ and $c_2 = 0.075$ due to coincident spikes of X_i and R within one time step. Spike trains of G_3 are highly correlate with each other with a CC of $c_3 = 0.2$. Here the CC c between two spike trains $x_i(t)$ and $x_j(t)$ is defined as $c = \langle (x_i(t) - \langle x_i(t) \rangle)(x_j(t) - \langle x_j(t) \rangle) \rangle / (\text{var}(x_i(t))\text{var}(x_j(t)))^{1/2}$ where $\text{var}(x_i(t))$ denotes the variance of $x_i(t)$. In this setup the inputs of the groups G_1 and G_2 carry relevant information whereas inputs of G_3 have "interesting" statistics (i.e. high correla-

[4] Here $\langle f \rangle_{Y|X} = \sum_Y f \cdot p(Y|X)$ denotes the expected value of f over Y conditioned on X.

[5] We define the rate of a spike train X_j at time step t as the current probability to spike $p(x_j^t = 1)$.

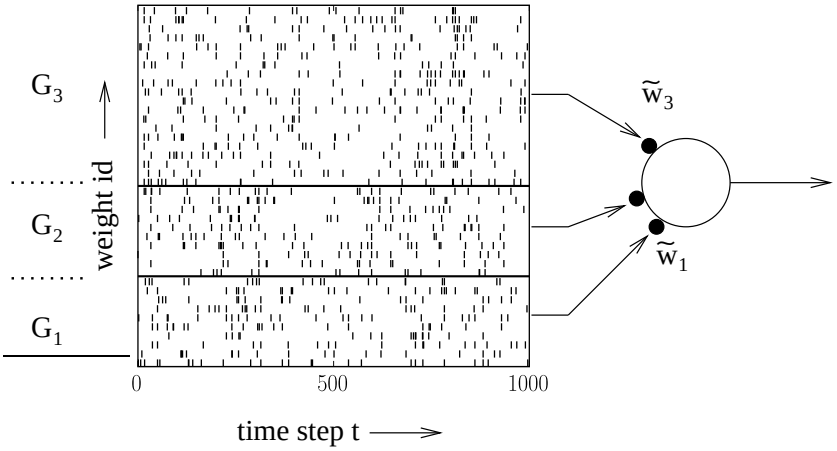


Figure 4.2: The setup of the simple IB task described in section 4.2.2. The synapses of the neuron are grouped into three groups G_1 to G_3 whose average weights are denoted as $\tilde{w}_1$ to $\tilde{w}_3$. The inputs to the neuron, illustrated here by a spike raster plot (notice that only 2/5 of the spike trains of each group are shown), as well as the relevance signal are modeled as spike trains. The different groups convey different amounts of relevant information in their precise spike timings, parametrized by the correlation coefficient between the input and the relevance signal. The IB learning rule (4.6) adapts the weights $\boldsymbol{w}$ such that eventually the output of the neuron is most informative about the relevance signal (with regularization).

tion) but which nevertheless are irrelevant due to the definition of R. Further simulation parameters and details can be found in appendix D.2.1. The preprocessing $\boldsymbol{h}^t = (h_1^t, h_2^t)$ was chosen in the following way. The first element $h_1^t = 1$ is a constant bias whereas $h_2^t = \sum_{s=0}^{\infty} \exp(-s/\tau) R^{t-s}$ is a low-pass filter of the relevance spike train with an exponential window of size $\tau = 10$.

In Fig. 4.3 the results of a simulation of this setup with a trade-off parameter $\gamma = 8 \cdot 10^{-6}$ are plotted. Fig. 4.3A shows the temporal evolution of the average group weights $\tilde{w}_a = |G_a|^{-1} \sum_{i \in G_a} w_i$ of group G_a for $a = 1, 2, 3$ with group size $|G_a|$. It can be observed that the average group weights $\tilde{w}_a$ converge to a value roughly proportional to the CC between the corresponding input spike trains and the relevance signal, e.g. the weights of G_1 are strongest after the learning. The inputs of G_3 do not carry relevant information by construction and therefore the weights decay towards zero due to the regularization term of the objective function (4.3). The dynamics of the parameters $\boldsymbol{q}$ are plotted in Fig. 4.3B. They eventually assume stationary values which are (possibly locally) optimal values for estimating the output probability $p(y^t = 1|R)$ conditioned on the relevance signal R via the estimator $F^t = \sigma(\boldsymbol{q}^t \cdot \boldsymbol{h}^t)$. An estimation of the IB objective function L_{IB} (for details see appendix D.2.1) and of the lower bound L are shown in Fig. 4.3C. Both measures increase over time due to the stochastic gradient ascent learning. Furthermore, L is quite "tight" for this task and provides the neuron with a good estimation of the "true" value of the objective function L_{IB} as well as its

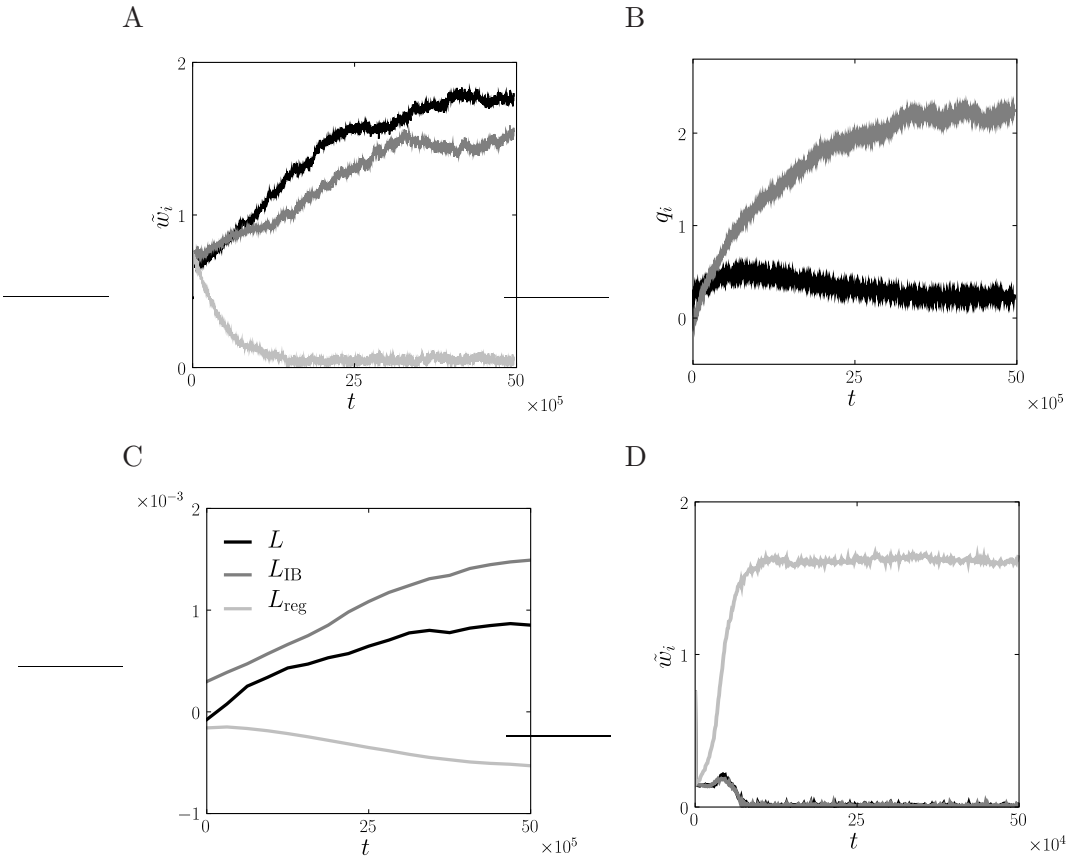


Figure 4.3: The results of the simulation described in section 4.2.2. A: The trajectories of the average group weights $\tilde{w}_i$ for $i = 1, \ldots, 3$ as a function of the time step t. The weights of G_1 (black) and G_2 (gray) are increased as they have non-vanishing mutual information with the relevance signal R. The weights of the remaining group G_3 (light gray) decay to zero as the corresponding inputs are independent of R. B: The trajectories of the parameters q_1 (black) and q_2 (gray). They evolve such that $\sigma(\boldsymbol{q}^t \cdot \boldsymbol{h}^t)$ optimally estimates the conditional probability $p(y^t = 1|R)$ (in terms of the Kullback-Leibler divergence). **C**: Numerical estimations of the IB objective function L_{IB} (gray) and of the lower bound L (black) are plotted as functions of the time step t. One sees that the lower bound gives a good estimation of L_{IB} in this example task. Furthermore, the regularization term L_{reg} is plotted (light gray). D: Results of applying an InfoMax learning rule to the same setup. InfoMax does not take the relevance signal R into account and therefore weights of group G_3 get potentiated (color coding as in panel A).

gradient wrt. $\boldsymbol{w}$ and $\boldsymbol{q}$. Additionally the regularization L_{reg} is plotted separately to illustrate its contribution to the total objective function L.

In Fig. 4.3D we show results of a simulation using the same setup as described above with the only difference that the weights are not learned with the IB learning rule but with an InfoMax learning rule. InfoMax aims at maximizing the amount of transmitted information $I(y^0, X^{-\infty})$ between input and output of the neuron without taking the relevance signal R into account. It can be seen that in contrast to the results of IB learning InfoMax potentiates the weights of G_3 as their inputs exhibit the strongest correlation with each other. The InfoMax rule is given in Appendix D.1.2 and a more general comparison to IB learning can be found in the Discussion section 4.4.

4.2.3 Neural Implementation of the Relevance Signal Preprocessing

In section 4.2.2 a simple IB optimization task was solved assuming that the filters $h = (h_1, \ldots, h_r)$ provide a suitable preprocessing of the relevance signal R (in that case a low pass filter of the relevance signal and a constant bias). In this example the preprocessing was quite specific for the given IB task (i.e. specific for the distribution $p(X^{-\infty}, R)$), and one might argue that the neuron could not have solved other IB tasks with this preprocessing. In this section, we address this point by proposing the implementation of the relevance signal preprocessing by a generic neural circuit, that is not tailored for a single IB task, but which allows the bottleneck neuron to solve a larger class of IB tasks with the same preprocessing filters. These tasks may also feature more complex statistically dependencies between the neural input X and the relevance signal R, in particular in the temporal domain.

In the approach for IB optimization presented above, it is essential for the bottleneck neuron to have a reasonable approximation F^t of the conditional probability $p(y^t = 1|R)$ in order to optimize the weights $\boldsymbol{w}$. The quality of the lower bound L which is optimized (i.e. the difference $|L_{\text{IB}} - L|$), is given by the Kullback-Leibler divergence between the estimation $F^t = \sigma(\boldsymbol{q} \cdot \boldsymbol{h}^t)$ and the "true" value $p(y^t = 1|R)$. Hence $|L_{\text{IB}} - L|$ critically depends on the given preprocessing h of R. Ideally the preprocessing would be powerful enough such that $F^t = p(y^t = 1|R)$ for some parameters $\boldsymbol{q}$, and optimizing L would then be equivalent to optimizing the "true" IB objective function L_{IB}. In (Maass, Natschlaeger, & Markram, 2002) the authors investigate the general problem of approximating with a fixed set of preprocessing filters (here $h = (h_1, \ldots, h_r)$) and a fixed class of memoryless readout functions (here the linear maps parametrized by $\boldsymbol{q}$ followed by the logistic function σ) any given target filter (here $p(y^t = 1|R)$)[6]. A largely positive result is given for approximating target filters that are time-invariant (TI) and that have the fading memory (FM) property (for exact definitions and results see (Maass et al., 2002)) under suitable assumptions concerning the set of filters and the set of readout maps. Roughly speaking a filter has fading memory if it becomes asymptotically insensitive to the remote history of its input. A further specific result given in (Boyd & Chua, 1985) states that TI-FM filters can be approximated with

[6]The considerations in (Maass et al., 2002) focus on the case of continuous time, but similar results also hold for discrete time.

arbitrary precision by a finite dimensional, linear dynamical system implementing the filters and a polynomial readout map.

Based on the theoretical results of (Maass et al., 2002), in a series of publications (for a review see (Buonomano & Maass, 2009), for a similar approach see (Jaeger, 2001)) it was observed that various TI-FM filters can be efficiently approximated using a fixed generic neural network implementing the filters h and exclusively learning a memoryless *linear* readout function. This approach exploits that sufficiently large recurrent networks of nonlinear neurons provide a sufficiently generic nonlinear preprocessing. Hence it often suffices to use linear, rather than polynomial, adaptive readouts. More precisely, in this approach the filters h are implemented by a sufficiently complex recurrent neural network which is generated randomly (in particular the network is not designed for approximating a specific filter) and which receives an external input given by the signal on which the target filter operates on (here the relevance signal R). The value $\boldsymbol{h}^t$ of the filters h at time step t is then defined e.g. as the vector of neuron activations (for continuous networks) or the output spikes of the network units at time step t. The readout map in these studies was restricted to linear maps, i.e. $\boldsymbol{q}^t \cdot \boldsymbol{h}^t$ and only the parameters $\boldsymbol{q}$ are learned in order to approximate the given specific target filter. This neural architecture poses a sensible implementation of the preprocessing h in the IB setup if the target filter $\sigma^{-1}(p(y^t = 1|R))$ can be assumed to have the FM property (it is guaranteed to be TI if R and X are stationary processes). The FM property amounts to assuming that input spikes x_i^t become asymptotically independent from the relevance signal $R^{t\pm\tau}$ for large delays τ. In the following paragraph an example IB task is discussed that illustrates this approach. We show that in this example a generic recurrent network with a trainable linear readout provides the bottleneck neuron with a sufficiently accurate estimation $F^t = \sigma(\boldsymbol{q} \cdot \boldsymbol{h}^t)$ of $p(y^t = 1|R)$ and hence allowing it to solve a given IB task.

Consider the following setup. Let the relevance signal R be a piece-wise constant, real-valued stochastic process which assumes every 30 time steps a new value which is iid.[7] in $[-0.5, 0.5]$, see Fig. 4.4A. The input spike trains X_i are grouped into four subgroups G_1 to G_4 similar to the setup of the example given in section 4.2.2. The inputs of the group G_j were generated as spike trains with a time-varying rate λ_j^t at time step t. The rate λ_1^t was defined as:

$$\lambda_1^t = aR^{t-\tau_1}R^{t-\tau_2} + b, \tag{4.7}$$

with delays $\tau_1 = 10$ and $\tau_2 = 50$ time steps and coefficients a, b. The remaining rates λ_2^t, λ_3^t, λ_4^t were generated with the same statistics as λ_1^t but they are independent from R. By construction only inputs of G_1 contain relevant information whereas the remaining inputs do not. The preprocessing of the relevance signal was implemented by a recurrent network of $r = 200$ sigmoidal rate neurons which receives the relevance signal R as input, i.e. the values of the filters $\boldsymbol{h}^t$ at time step t were chosen as the network activity at time step t. According to the approach proposed above, the estimation F^t was given by $\sigma(\boldsymbol{q} \cdot \boldsymbol{h}^t)$ and the parameters $\boldsymbol{q}$ were learned by (4.6). The quantity F^t

[7] iid. = identically, independently distributed

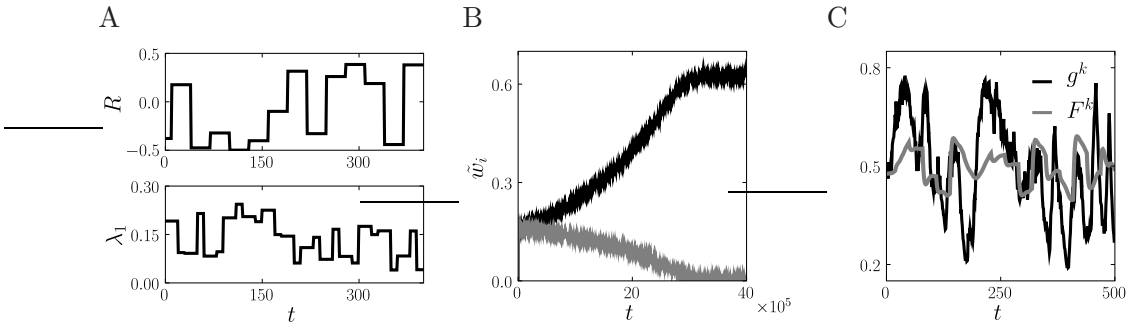


Figure 4.4: Numerical results for an IB optimization task with a preprocessing of the relevance signal R implemented by a generic recurrent network as described in section 4.2.3. A: Shown is the relevance signal R (top) as well as the spiking probability λ_1 (bottom) for the inputs of group G_1. By design, only λ_1 is statistically dependent on R and hence only the inputs of G_1 carry relevant information. B: The trajectories of the mean weights $\tilde{w}_1$ (black) of G_1 and $\tilde{w}_\not{1}$ (gray) of the remaining groups G_2, G_3, G_4 are plotted as functions of time step t. As only the inputs of G_1 have a non-vanishing mutual information with R, $\tilde{w}_1$ is exclusively potentiated. The average $\tilde{w}_\not{1}$ of the remaining weights decays due to regularization. C: Trajectories of the activation function $g^t = p(y^t = 1|X^{-\infty,t})$ (black) and of the estimator F^t (gray) are shown for an interval of 500 time steps after the weights $\boldsymbol{w}$ and the parameters $\boldsymbol{q}$ have been learned.

can be interpreted as the activity of a logistic readout neuron with input $\boldsymbol{h}^t$ from the recurrent network and weights $\boldsymbol{q}^t$. All further details and parameter can be found in appendix D.2.2.

The simulation results of the average weights $\tilde{w}_1 = \frac{1}{25}\sum_{j\in G_1} w_j^t$ of G_1 and $\tilde{w}_\not{1} = \frac{1}{75}\sum_{j\notin G_1} w_j^t$ of the remaining groups G_2, G_3, G_4 are presented in Fig. 4.4B. In agreement with the learning goal, only the weights from G_1 are potentiated, while the other weights decay to zero due to the regularization term in the objective function L defined in (4.3). In Fig. 4.4C the spiking probability $g^t = p(y^t = 1|X^{-\infty,t})$ and the estimation $F^t \approx p(y^t = 1|R)$ are plotted for 500 time steps after learning of $\boldsymbol{w}$ and $\boldsymbol{q}$. Although the rates of G_1 are related to the relevance signal R by a second order Volterra series defined in (4.7) which involves temporal delays of 10 and 50 time steps, the estimation F^t is sufficiently accurate for learning an approximate IB optimal coding. Hence, this task is an example where the preprocessing of the relevance signal R via a generic untrained recurrent network with a trainable readout enables the neuron to extract statistical dependencies between $X^{-\infty}$ and R and to solve the given IB task.

4.3 Application: Predictive Coding

In (Bialek et al., 2006) the H1 neuron of the blowfly, an extensively studied cell of the fly sensory-motor control system, is proposed as a possible example for a biological system providing an IB optimal coding. It is hypothesized that the output of this neuron is maximally informative about *future* external stimuli, hence this coding paradigm is

termed *predictive coding*. In the following section we show how such a predictive coding scheme can be learned by a neuron using a variant of the presented IB learning rule (4.6).

It is has often been hypothesized that biological agents maintain an internal representation, denoted here as X_{int}, of the external world which is obtained and updated via previous sensory stimuli X_{past} (as sensing is a causal process which takes time). The representation X_{int} allows the agent to adapt its behavior to the state of the environment and plan future actions. The hypothesis that this representation X_{int} is optimal in some information theoretic sense (for a given amount of invested resources L_{reg}) has drawn much attention and served as a guideline for many intriguing studies (see references in (Bialek et al., 2006)). In (Bialek et al., 2006) however, the authors argue that not all sensory information contained in X_{past} is equally important for the behavior and the survival capabilities of the agent and hence not the entire sensory information should be represented internally in X_{int}. More precisely, it is hypothesized that only such external stimuli are worth being represented that are informative about the future sensory stimuli X_{future} which encode the future state of the environment. Only this information can be used by the agent to plan behavior and eventually improve its fitness. This predictive coding paradigm was formalized as an IB optimization problem: The mapping from the past sensory stimuli X_{past} to the representation X_{int} should be chosen such that it maximizes the predictive information $I(X_{\text{int}}, X_{\text{future}})$ about the future sensory stimuli X_{future} at fixed costs L_{reg}. Following this train of thought, the agent should hence maximize the following IB objective function $L_{\text{predictive}}$:

$$L_{\text{predictive}} = I(X_{\text{int}}, X_{\text{future}}) - \gamma L_{\text{reg}}.$$

The authors of (Bialek et al., 2006) also discuss a concrete example of this predictive coding paradigm, the H1 neuron of the blowfly. This neuron is part of the optomotor control loop and it is known to approximately code logarithmically for the horizontal angular velocity of the fly. In the study (Bialek et al., 2006) it is argued that this specific coding of the H1 neuron of the external stimuli could be optimal wrt. the objective function $L_{\text{predictive}}$.

Here we show that a single neuron can learn to extract predictive information from its inputs and establish a predictive coding scheme, similar to the one described in (Bialek et al., 2006), using a slightly modified version the IB learning rule (4.6). At any time step t, we identify the sensory input X_{past} with the input history $X^{-\infty,t}$ of the bottleneck neuron and we identify the internal representation X_{int} with its output y^t. Furthermore, we define the relevance signal X_{future} as the future input $X^{t,t+\delta}$ to the neuron in the time interval $[t, t+\delta]$, which extends δ time steps into the future for a given parameter $\delta \in \mathbb{N}$. For the sake of simplicity we also assume in this section that the activation function $g = \sigma$ of the bottleneck neuron is the logistic function[8].

One possible approach to learn a predictive code is the following. If we assume that the synaptic kernel ϵ (see eqn. (4.1)) is non-anticipating and that its support is shorter

[8]Similar learning rules can also be derived for more general activation functions, but they turn out to be slightly more complex.

than δ time steps, the future activation $g^{t+\delta}$ is exclusively a function of the future input $X^{t,t+\delta}$. Hence $g^{t+\delta}$ can be interpreted as a preprocessing $\boldsymbol{h}^t$ of the relevance signal $X^{t,t+\delta}$. Based on this observation we make the ansatz $F^t = g^{t+\delta}$ for the estimator F^t, i.e. we hypothesize that the neuron uses its own future spiking probability $g^{t+\delta}$ to estimate the amount of predictive (=relevant) information contained in its input at time step t. The objective function L to maximize resulting from this approach reads:

$$L = \left\langle \log \left(\frac{(g^\delta)^{y^0}(1-g^\delta)^{y^0}}{p(y^0)} \right) \right\rangle - \frac{\gamma}{2}\boldsymbol{w}^2. \tag{4.8}$$

Due to the ansatz $F^t = g^{t+\delta}$ the objective function (4.8) and consequently its gradient at time step t contains the spiking probabilities g^t and $g^{t+\delta}$. Performing a straight forward stochastic gradient ascent (analogously to the procedure that lead to the rule (4.6)) would result in an anticipating learning rule, i.e. the weight update $\Delta\boldsymbol{w}^t$ would involve the future spiking probability $g^{t+\delta}$. This can be circumvented by shifting the time step index on the rhs of the learning rule by $-\delta$, which is allowed in stochastic gradient ascent as this does not change the expected value of the learning rule. This leads to the following update equation for the weights:

$$\eta_w^{-1}\Delta\boldsymbol{w}^{t+1} = g'^{t-\delta}\boldsymbol{\nu}^{t-\delta}\Big(\sigma^{-1}(g^t) - \sigma^{-1}(\langle g^{t-\delta}\rangle)\Big) - \gamma\boldsymbol{w}^t + \boldsymbol{\nu}^t(g^{t-\delta} - g^t). \tag{4.9}$$

The above learning rule is non-anticipating but it is still not local in time as it contains the terms $g^{t-\delta}$, $g'^{t-\delta}$ and $\boldsymbol{\nu}^{t-\delta}$. Therefore the values of the activity g^t as well as the synaptic activity $\boldsymbol{\nu}^t$ have to be buffered by the neuron for δ time steps in order to learn a predictive code with the rule (4.9). Although an exact implementation of this buffering seems rather implausible, approximate implementations of the predictive coding learning rule might be biologically achievable. Assuming that the time parameter δ is not exactly defined but is rather given by a more diffuse parameter range, running averages of g and ν with appropriate window sizes might prove to be sufficiently informative in order to learn an approximate predictive code. These averages could possibly be encoded in the signaling cascades that are triggered by pre- and postsynaptic spike events.

The structure of the learning rule (4.9) is similar to the one of the general IB rule (4.6). The first term, which is proportional to $\boldsymbol{\nu}^{t-\delta}\sigma^{-1}(g^t)$, potentiates those synapses whose input at time $t-\delta$ is correlated with the output rate g^t at time step t, i.e. those synapses are potentiated whose inputs are predictive for the future neural output. The next term, which is proportional to $\sigma^{-1}(\langle g^{t-\delta}\rangle)$, remains unchanged from the original rule (4.6). The last term $\boldsymbol{\nu}^t(g^{t-\delta} - g^t)$ stems from the fact that the estimator $F^t := g^{t+\delta}$ depends now on $\boldsymbol{w}$ itself. This term replaces the learning rule for $\boldsymbol{q}$ form (4.6) (second line), and it drives the weights such that the past activity $g^{t-\delta}$ can be well estimated by the present activity g^t.

We want to point out that the simple choice for the estimator $F^t = g^{t+\delta}$ made above limits the power of the rule (4.9) for learning a predictive code. Only those weights w_j are potentiated whose input X_j is positively correlated at time t with the output at time $t+\delta$. Negative correlations or higher order statistical dependencies cannot be extracted

with this choice of the estimator F. In order to achieve this, a more sophisticated ansatz for F with a more diverse preprocessing of $X^{t,t+\delta}$ would be required (e.g. the ansatz proposed in section 4.2.3).

We illustrate the behavior of the learning rule (4.9) by a simple numerical example for a delay parameter $\delta = 25$. Consider the following setup where the synapses are again divided into four groups $G_1, \ldots, G_4$. Synapses from subgroup G_1 receive spike trains with a rate that is determined by a (discrete time) Ornstein-Uhlenbeck[9] (OU) process with a time constant $\tau_1 = 50$, mean $\mu_1 = 0.2$ and standard deviation (SD) $\sigma_1 = 0.3 \cdot \mu_1$. The inputs for group G_2 are generated in a similar way, however with a time constant $\tau_2 = 25$ and $\sigma_2 = 0.5 \cdot \mu_1$. A (discrete time) telegraph process with mean μ_1 and SD σ_1 and a time constant $\tau_3 = 20$ determines the rate for the spike train of group G_3. Spike trains of G_4 are generated with a constant rate μ_1. Additional parameters and details can be found in appendix D.2.3. The result of the simulation are plotted in Fig. 4.5. As expected the weights of G_4 rapidly decay as they transmit no relevant information. Further, due to the long autocorrelation time constant τ_1 the weights of G_1 are exclusively potentiated while the weights for G_2 and G_3 decay. If however the values of the time constants τ_1 and τ_3 are switched ($\tau_1 = 20$, $\tau_3 = 50$) the results are reversed, i.e. weights of G_3 grow over time while those of G_1 decay (results not shown). This example illustrates that the specialized version (4.9) of the IB rule (4.6) enables the neuron to extract predictive information from its input in simple setups.

4.4 Discussion

4.4.1 Relation to Existing Work

Here we briefly discuss existing work that is related to the IB learning rules proposed in this contribution. Additionally, in the first paragraph the differences between the approach presented in this chapter and other IB algorithms are described.

Other IB Algorithms

Most IB algorithms determine nonparametric mappings from the input to the output RVs e.g. (Tishby et al., 1999; Slonim & Tishby, 1999). Hence, the approach presented here, which determines an IB optimal mapping via gradient ascent wrt. to the model parameters $\boldsymbol{w}$, $\boldsymbol{q}$ might be regarded to be against the spirit of the IB framework. However we argue that in the considered neural setup, where the global structure of the IB mapping is fixed (e.g. the dimensionality of the output and the class of transformations that can be used), such a parametric approach is nevertheless justified. Another difference is that most IB algorithms operate in batch mode on the complete input data (with the notable exceptions of predictive coding described below), whereas the setup we propose maps input sequences onto an output sequence *online*. This approach reflects the fact that neurons naturally operate in the temporal domain, which also requires an online algorithm for learning the IB optimal mapping. Furthermore, most IB algorithms

[9]More precisely the firing rate $p(x_i(t) = 1)$ is defined as $p(x_i(t) = 1) = \min\{1, \max\{0, \mathrm{O}(t)\}\}$ in terms of the OU process $\mathrm{O}(t)$.

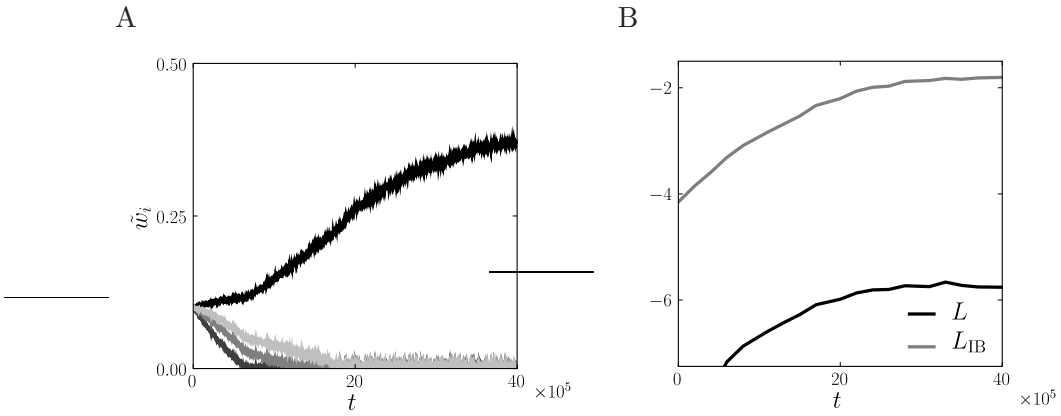


Figure 4.5: The numerical results for the predictive coding application described in section 4.3. A: Shown are the average group weights $\tilde{w}_i$ for G_1 (top curve in black), G_2 (dark gray), G_3 (gray) and G_4 (light gray). The group G_1 transmits the largest amount of predictive information due to the long autocorrelation time constant τ_1 of its input and hence the average $\tilde{w}_1$ is exclusively increased whereas the remaining weights decay to zero. B: The lower bound L (black) is plotted as a function of the time step t. Also shown is an estimation of the "true" IB objective function $L_{\mathrm{predictive}}$ (gray), for which the mutual information $I(y^t, X^{t,t+\delta})$ was approximated by $\sum_{j=1}^{N} I(y^t, X_j^{t,t+\delta})$ (causing the large offset between L and $L_{\mathrm{predictive}}$). The trajectories indicate that the neural output becomes more predictive for the future input $X^{t,t+\delta}$.

assume that the joint distribution $p(X, R)$ of the input RV X and the relevance RV R is known, hence they require a beforehand estimation of $p(X, R)$ based on finite samples (for an in-depth analysis of this procedure see (Shamir, Sabato, & Tishby, 2008)). In contrast to this procedure, our approach (based on the objective function (4.3)) directly unifies this estimation process and learning of the IB mapping. This unification however comes at the expense of not optimizing the "true" IB objective function L_{IB} but a lower bound L of the latter.

The work presented here directly builds on (Klampfl et al., 2009). There, the authors derived an online IB learning rule by gradient ascent for a quite sophisticated stochastic neuron model assuming that the relevance signal sequence is given by a spike train of the same neuron model. The gradient of the relevant information (the mutual information between the output Y and the relevance signal R) was estimated by measuring the correlation $\langle y^t R^t \rangle$, where $R^t \in \{0, 1\}$ is the relevance spike train at time step t. The resulting learning rule is only sensitive to instantaneous correlations between the neural output and the relevance signal, a fact that limits the applicability of the learning rule. In contrast, we propose in this study the more general approach of a parametric estimation of the gradient of $I(y^0, R)$ based on a given preprocessing $\boldsymbol{h}^t = (h_1^t, \ldots, h_r^t)$. The resulting rule (4.6) is as powerful as the preprocessing that is available to the neuron. Given that neurons are strongly interconnected and receive many of recurrent inputs resulting in highly nontrivial transformations of the external input, it seems reasonable to assume that the preprocessing of the relevance signal, that is potentially available

to the bottleneck neuron, is diverse and "rich" enough to carry out a large class of IB optimization tasks. Further, a considerable simplification of the learning rule (4.6) compared to the one presented in (Klampfl et al., 2009) was achieved by choosing to maximize the mutual information $I(y^0, R)$ instead of the more complex quantity $I(Y, R)$, where $Y = (\ldots, y^{-1}, y^0, y^1, \ldots)$ [10].

InfoMax and Imax

The learning goal of maximizing the mutual information between input and output of individual neurons or neural networks, so-called InfoMax learning, has served as a fruitful theoretical principle for learning with artificial and more biologically realistic neural models, see e.g. (Linsker, 1989; Bell & Sejnowski, 1995; Chechik, 2003; Toyoizumi et al., 2005; Parra et al., 2009). More precisely InfoMax is defined as learning a neural mapping $X \to Y$ of some input X to the output Y which maximizes the amount of transmitted information defined as the mutual information $I(X, Y)$ (with some regularization constraints). While InfoMax shares a common theoretical foundation with the IB method, namely information theory, there are differences wrt. the specific learning goals and their biological interpretation. InfoMax is an unsupervised learning principle, its formulation only involves the input X and the output Y. There is no external "guideline" of how the input X is to be transformed into the output Y except for maximizing the scalar mutual information $I(X, Y)$. InfoMax can be interpreted as a possible approach to dimensionality reduction techniques, to clustering as well as to blind source separation (Bell & Sejnowski, 1995). The IB method also aims at constructing a mapping $X \to Y$ of the input X to the output Y which exhibits certain information theoretic properties. In contrast to InfoMax however, the IB method is not an unsupervised learning framework. In the IB framework it is assumed that the environment offers information about what can be considered relevant in the input via the given relevance signal R. It has to be emphasized that the IB mapping, once learned, maps the input to the output $X \to Y$ and it is *not* a mapping from the input and the relevance signal to the neural output $X \times R \to Y$. Thus, relevant information given by R which is not present in the input X will not be encoded in the output Y. Further it should be noted that the relevance signal R only has to be present during learning of the IB mapping. After this learning phase the IB optimal mapping $X \to Y$ can be carried out by the neural architecture without the presence of the relevance signal R. In a biologically plausible setting the distinction between learning and operation phase could e.g. be implemented with a learning rate $\eta(\|R^t\|)$ that detects the presence of the relevance signal by monitoring some measure of its intensity $\|R^t\|$ over time and which stops learning if the relevance signal is absent.

To further illustrate the difference between IB and InfoMax consider the setup of the simulation presented in section 4.2.2. The input to the IB neuron consists of three groups G_1, G_2 and G_3. The results presented in 4.2.2 show that if the relevance signal R is statistically dependent on the input G_1 and G_2, i.e. G_1, G_2 convey relevant information, the corresponding weights are potentiated, see Fig. 4.3A. If however the synapses are

[10] This simplification can apparently be made without reducing the power of the learning rule as all numerical IB task presented by (Klampfl et al., 2009) can also be solved by the rule (4.6) even when assuming only a simple preprocessing (data not shown).

updated with an InfoMax learning rule (for details see appendix D.1.2) only the weights of group G_3 are potentiated while all other weights decay, see Fig. 4.3D. This weight configuration maximizes the transmitted mutual information $I(X, Y)$ since the group G_3 subsumes the most afferents and its inputs exhibit the strongest spike-spike correlations. This is an example where the learning results of IB and InfoMax differ.

(Becker, 1996; Becker & Hinton, 1992) propose a learning principle called Imax which is similar to IB learning and can be interpreted as a special case of the latter. The objective of Imax is to maximize the mutual information between the *outputs* of two (or more) networks which receive disjoint but statistically related inputs. It is therefore different from InfoMax which aims as maximizing the mutual information between input and output. More precisely in (Becker, 1996) two (multi-layer) feedforward networks are considered, whose two inputs X_1, X_2 are given by neighboring patches of visual input. The learning objective was defined as maximizing the mutual information $I(\hat{X}_1, \hat{X}_2)$ between the activations $\hat{X}_1$, $\hat{X}_2$ of the output layers of the networks. Partial derivatives of $I(\hat{X}_1, \hat{X}_2)$ are propagated back into the hidden layers of the network to maximize $I(\hat{X}_1, \hat{X}_2)$. In contrast to the work presented in this chapter, the architecture of (Becker, 1996; Becker & Hinton, 1992) does not operate in the temporal domain, i.e. it implements an instantaneous mapping from input to output (which simplifies drastically the evaluation of the objective function $I(\hat{X}_1, \hat{X}_2)$). An interesting topic for future research is to port the architecture proposed in (Becker, 1996; Becker & Hinton, 1992) to neurons operating in time by using the learning rule presented here. This can possibly be achieved by adopting ideas from the symmetric IB setup presented in (Friedman, Mosenzon, Slonim, & Tishby, 2001), where two disjoint input streams are mapped to simpler representations while preserving as much mutual information between each other.

Predictive Coding

Predictive coding, which was formalized in the IB framework in (Bialek et al., 2006), has been studied for linear mappings and Gaussian noise in (Creutzig & Sprekeler, 2008), revealing an intriguing relation to slow feature analysis (for an introduction see (Wiskott & Sejnowski, 2002)). The solutions to this past-future bottleneck are explicitly given. Furthermore, the analysis of predictive coding was expanded to linear dynamical systems in (Weiss, 2007) also resulting in a complete characterization of the IB optimal systems assuming a linear dependence of the input RV X on the relevance RV R with additive Gaussian noise. These studies provide strong, exact results to the considered restricted setups. The spirit of the approach presented here is quite different. We provide an iterative scheme for IB optimization, which is possibly prone to local minima, focusing on a neural mapping while making only few assumptions about the input and relevance processes X and R.

Predictive coding as a learning goal for sensory processing with neural architectures was motivated in (Bialek et al., 2006) by arguing that this paradigm allows to learn an "useful" (wrt. the agent's fitness) internal representation or model of the environment. It can therefore be considered to be closely related to learning a generative model of the environment (see (Slonim & Weiss, 2003) for a relation between IB and generative models). However it can be argued that learning a sufficiently accurate model of the

environment may consume too many resources and may require too much data to be a suitable strategy for adapting to the environment. An alternative would be a discriminative approach, i.e. one might hypothesize that it is more appropriate for an agent to directly learn a mapping from environmental configurations to behavioral decisions, without the need for an explicit representation of the environment. Which of these two approaches, generative versus discriminative, is the better theoretical model depends amongst others on the structure and amount of data that the agent learns from. In a machine learning context (Hinton, 2007) argues that a combination of generative learning, making use of unlabeled data, and discriminative learning is a powerful and promising approach. This indicates that such a combination of discriminative and generative approaches might also be a powerful model for sensory processing in biological agents.

4.4.2 A Possible Biological Implementing of IB Optimization with Spiking Neurons

The experimental investigation of synaptic plasticity has made significant advances in the last decade (for reviews see (Caporale & Dan, 2008; Sjöström, Rancz, Roth, & Häusser, 2008)). The classical picture of synaptic plasticity, as postulated by (Hebb, 1949) and later experimentally described by others, which exclusively depends on the pre- and postsynaptic activity had to be considerably expanded over the years due to accumulating experimental evidence. It is now known that many additional factors modulate synaptic weight changes, e.g. neuromodulators (Hee et al., 2007), details of the neural morphology (Sjöström et al., 2008) and extracellular subthreshold stimulation (Sjöström et al., 2001). This large body of experimental literature poses a huge challenge for theoretical work about the underlying functions of these mechanisms for neural information processing. The fact that synaptic plasticity is determined by additional quantities besides pre- and postsynaptic activity might be a suitable mechanism allowing synaptic weight changes to fulfill complex optimization goals like IB optimization. In the following we show that the proposed IB learning rule (4.6) fits well to recent experimental results concerning the influence of dendritic depolarization on synaptic plasticity.

The plasticity of synapse j described by the rule (4.6) depends on the synaptic activity ν_j^t, in agreement with experimental findings. Further a measure of the average postsynaptic activity $\langle g^t \rangle$ influences the weight change. This may be interpreted in a biological context as a homeostatic control of the weights. The central claim of the learning rule (4.6) is however that a "third" factor $\sigma^{-1}(F^t)$, which quantifies the influence of the relevance signal, shapes synaptic plasticity. This contribution crucially determines the sign and amplitude of the weight change in this plasticity model. A biological mechanism termed "dendritic switch", which has recently been uncovered in (Sjöström & Häusser, 2006), is a plausible candidate for a third factor modulating plasticity as required by IB learning. It is known that plasticity of dendritic synapses depends on the backpropagating action potential (bAP) in the dendrite (Caporale & Dan, 2008). Further it has been shown that the bAP amplitude and the reliability of the bAP are shaped by the active and passive conductance properties of the dendrite, see (Stuart & Häusser, 2001).

These conductance properties can in turn be considerably modulated by local de- or hyperpolarization of the dendrite as demonstrated in (Stuart & Häusser, 2001). Hence it can be assumed that properly timed EPSPs/IPSPs in the dendrite shape synaptic plasticity by influencing the bAP amplitude. According to (Sjöström & Häusser, 2006), these mechanisms indeed enable proximal synapses to act as "dendritic switches" which modulate the weight changes at distal synapses via boosting or shunting the bAP. These "dendritic switches" were shown to be able to change the amplitude as well as the sign of the weight change at distal synapses by modulating the bAP with EPSPs/IPSPs that de-/hyperpolarize the proximal part of the dendrite. In the presented IB model the proximal synapses, the dendritic switches, would convey the influence of the relevance signal $\sigma^{-1}(F^t)$, see Fig. 4.6. The weights $\boldsymbol{w}$, which obey the rule (4.6), would correspond to more distal synapses whose plasticity is controlled by the relevance signal. With this correspondence the IB plasticity model predicts a boosting/shunting of bAPs leading to potentiation/depression at active distal synapses (those with $\boldsymbol{\nu}^t > 0$) whenever the weighted, preprocessed relevance signal $\sigma^{-1}(F^t)$ (representing the input at the proximal synapses) is high/low. As the dendritic switches act on a millisecond time scale this mechanism would provide a sufficiently high temporal resolution for the relevance signal in contrast to other factors modulating plasticity (e.g. neuromodulators).

In spite of this possible correspondence with the experimental data discussed above we wish to point out the limitations of the IB learning rule (4.6) as a theoretical model for experimental findings. It has to be noted that the IB learning rule presented here cannot account e.g. for the effect of spike timing-dependent plasticity (STDP) as e.g. reported in (Bi & Poo, 1998). The weight change given by the IB rule (4.6) does not exhibit a dependence on the postsynaptic spike times, only on a long-term average of the postsynaptic firing rate $\langle g \rangle$. This is in contrast to the experimental results on STDP which report a strong dependence of plasticity on the precise timing of pre- and postsynaptic spikes. Furthermore numerous more subtle aspects of plasticity, such as postsynaptic voltage dependence and weight dependence of plasticity, are not reflected by the IB rule. It is the topic of current research whether the IB approach in conjunction with more realistic neuron models or other constraints can reproduce experimental data more faithfully.

4.4.3 Summary

In this chapter we presented an online learning rule for IB optimization with a simple, idealized spiking neuron model. The neuron was regarded as an information bottleneck that maps its high dimensional input sequence on a one dimensional output sequence of spikes. By the help of the proposed IB learning rule the neuron can adapt its weights such that its output contains the maximal amount of relevant information, i.e. its output is maximally informative (possibly locally optimal) about a relevance signal also given by a sequence of RVs in time. This learning rule was derived assuming that the neuron has access to an estimation of its currently transmitted amount of relevant information (more precisely the gradient), which is based on a given preprocessing of the relevance signal and an adaptable set of parameters which are learned simultaneously with the

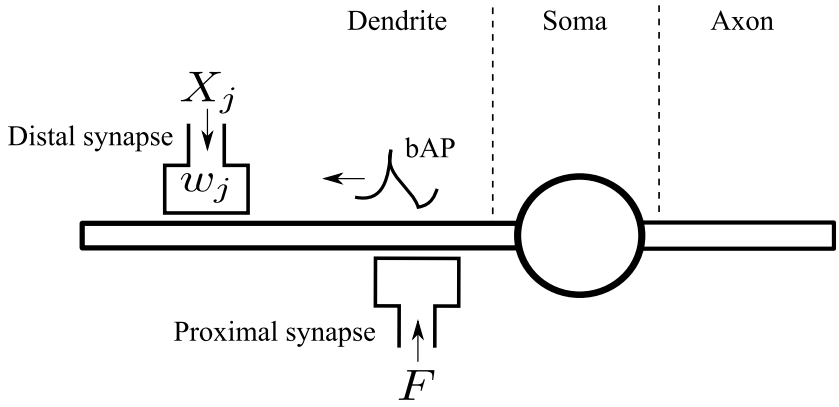


Figure 4.6: A possible biological mechanism implementing IB optimization in a single neuron. As shown in experiments, the activity of proximal synapses can act as "dendritic switches" that critically influence the amplitude and the sign of weight changes of distal synapses. The IB learning rule could be implemented in the distal synapses assuming that the proximal synapses convey the influence of the relevance signal R via F^t.

weights. This approach extends previous studies on neural IB optimization (Klampfl et al., 2007, 2009) that were based on correlations between the output of the bottleneck neuron and the relevance signal.

We also addressed the question of how a suitable and sufficiently general preprocessing of the relevance signal may be implemented in a biological neural system. Motivated by previous theoretical, numerical and experimental studies (see (Buonomano & Maass, 2009)) we argued that a generic recurrent neural circuit, which is not learned for a certain IB task and hence looks randomly structured from the perspective of the bottleneck neuron, can be considered a plausible candidate for such an implementation of the preprocessing. Simple models of recurrent neural networks were shown in previous studies to provide a considerable amount of memory and "nonlinearity" and hence render themselves to be a suitable preprocessing of the relevance signal enabling the bottleneck neuron to carry out a class of IB tasks.

Further, we have discussed a biological mechanism that can in principal resolve the problem of spacial non-locality encountered in previous IB learning rules for spiking neurons, i.e. the problem of how the current values of quantities that are essential to the learning rule can be made available at the location of the synapse. The recently discovered mechanism of dendritic switches (Sjöström & Häusser, 2006) seems well suited as an implementation of a learning process that is modulated by an external "third" factor (the relevance signal in in addition to the two factors given by pre- and postsynaptic signals) as required by IB learning.

Predictive coding has been proposed as an unsupervised learning goal for single neurons in (Bialek et al., 2006), a hypothesis that seems to be in agreement with experimental findings. As an application of IB learning with spiking neurons, we have shown that a variant of the proposed IB learning rule enables the neuron to learn a predictive code assuming simple input statistics. It has to be emphasized that several neural learning rules exist which extract temporal regularities from the input. In a recent study

(Creutzig & Sprekeler, 2008), a close relation between IB optimization and learning of temporal invariances in a more machine-learning oriented setting was pointed out. The approach presented here shows that also on a single neural level the well-known learning goals related to temporal invariance can be motivated and a viable learning rule can be derived from the IB framework.

The proposed learning rules are based on idealized assumptions especially wrt. the neuron model. The applied neuron model neglects several characteristics observed in experiments, most prominently a refractory mechanism, complex voltage dynamics (e.g. bursting, rebound spikes) as well as spacial extension and morphology of a neuron. We argue that studying learning in highly simplified system is nevertheless sensible as it possibly provides a "baseline" architecture, i.e. a possible learning strategy and its essential functional building blocks, that is not cluttered by (hopefully) unimportant contingent details of the neural dynamics.

4.5 Acknowledgments

This chapter is based on the paper *A spiking neuron as information bottleneck*, which was written by Lars Büsing (LB) and Wolfgang Maass (WM). The model was developped by LB. The simulations were designed by LB and WM and conducted by LB. LB wrote the manuscript.

Chapter 5

Computational Power and the Order-Chaos Phase Transition in Reservoir Computing

Contents

Randomly connected recurrent neural circuits have proven to be very powerful models for online computations when a trained memoryless readout function is appended. Such *Reservoir Computing (RC)* systems are commonly used in two flavors: with analog or binary (spiking) neurons in the recurrent circuits. Previous work showed a fundamental difference between these two incarnations of the RC idea. The performance of a RC system built from binary neurons seems to depend strongly on the network connectivity structure. In networks of analog neurons such dependency has not been observed. In this article we investigate this apparent dichotomy in terms of the in-degree of the circuit nodes. Our analyses based amongst others on the Lyapunov exponent reveal that the phase transition between ordered and chaotic network behavior of binary circuits qualitatively differs from the one in analog circuits. This explains the observed decreased computational performance of binary circuits of high node in-degree. Furthermore, a novel mean-field predictor for computational performance is introduced and shown to accurately predict the numerically obtained results.

5.1 Introduction

In 2001, Jaeger (Jaeger, 2001) and Maass (Maass et al., 2002) independently introduced the idea of using a fixed, randomly connected recurrent neural network of simple units as a set of basis filters (operating at the edge-of-stability where the system has fading memory). A memoryless readout is then trained on these basis filters in order to approximate a given time-invariant target operator with fading memory (Maass et al., 2002). Jaeger used analog sigmoidal neurons as network units and named the model

Echo State Network (ESN). Maass termed the idea Liquid State Machine (LSM) and most of the related literature focuses on networks of spiking neurons or threshold units. Both ESNs and LSMs are special implementations of a concept now generally termed Reservoir Computing (RC) which subsumes the idea of using general dynamical systems (e.g. a network of interacting optical amplifiers (Vandoorne et al., 2008)) – the so-called reservoirs – in conjunction with trained memoryless readout functions as computational devices. These RC systems have already been used in a broad range of applications (often outperforming other state-of-the-art methods) such as chaotic time-series prediction (Jaeger & Haas, 2004), single digit speech recognition (Verstraeten, Schrauwen, Stroobandt, & Campenhout, 2005), and robot control (Joshi & Maass, 2005).

Although ESNs and LSMs are based on very similar ideas (and in applications it seems possible to switch between both approaches without loss of performance (Verstraeten, Schrauwen, D'Haene, & Stroobandt, 2007)) an apparent dichotomy exists in the influence of the reservoir's topological structure on its computational performance. The performance of an ESN using analog, rate-based neurons, is e.g. largely independent of the sparsity of the network (Jaeger, 2007) or the exact network topology such as small-world or scale-free connectivity graphs[1]. For LSMs, which consist of spiking or binary units, the opposite effect has been observed. For the latter systems, the influence of introducing e.g. small-world or biologically measured lamina-specific cortical interconnection statistics (Haeusler & Maass, 2007) clearly leads to an increase in performance. In the results of (Bertschinger & Natschlaeger, 2004) it can be observed (although not specifically stated there) that for networks of threshold units with a simple connectivity topology of fixed in-degree per neuron, an increase in performance can be found for decreasing in-degree. None of these effects can be reproduced using ESNs.

In order to systematically study this fundamental difference between binary (spiking) LSMs and analog ESNs, we close the gap between them by introducing in Sec. 5.2 a class of models termed quantized ESNs. The reservoir of a quantized ESN is defined as a network of discrete units, where the number of admissible states of a single unit is controlled by a parameter called quantization level. LSMs and ESNs can be interpreted as the two limiting cases of quantized ESNs for low and high quantization level respectively. We numerically study the influence of the network topology in terms of the in-degree of the network units on the computational performance of quantized ESNs for different quantization levels. This generalizes and systemizes previous results obtained for binary LSMs and analog ESNs.

In Sec. 5.3 the empirical results are analyzed by studying the Lyapunov exponent of quantized ESNs, which exhibits a clear relation to the computational performance (Legenstein & Maass, 2007a). It is shown that for ESNs with low quantization level, the chaos-order phase transition is significantly more gradual when the networks are sparsely connected. It is exactly in this transition regime that the computational power of a Reservoir Computing system is found to be optimal (Legenstein & Maass, 2007a). This effect disappears for ESNs with high quantization level. A clear explanation of the influence of the in-degree on the computational performance can be found by investigating

[1]Shown by results of unpublished experiments which have also been reported by the lab of Jaeger through personal communication.

the rank measure presented in (Legenstein & Maass, 2007a). This measure characterizes the computational capabilities of a network as a trade-off between the so-called kernel quality and the generalization ability. We show that for highly connected reservoirs with a low quantization level the region of an efficient trade-off implying high performance is narrow. For sparser networks this region is shown to broaden. Consistently for high quantization levels the region is found to be independent of the interconnection degree.

In Sec. 5.4 we present a novel mean-field predictor for computational power which is able to reproduce the influence of the topology on the quantized ESN model. It is related to the predictor introduced in (Bertschinger & Natschlaeger, 2004), but it can be calculated for all quantization levels, and can be determined with a significantly reduced computation time. The novel theoretical measure matches the experimental and rank measure findings closely.

5.2 Online Computations with Quantized ESNs

We consider networks of N neurons with the state variable $\mathbf{x}(t) = (x_1(t), \ldots, x_N(t)) \in [-1, +1]^N$ in discrete time $t \in \mathbb{Z}$. All units have an in-degree of K, i.e. every unit i receives input from K other randomly chosen units with independently identically distributed (iid.) weights drawn from a normal distribution $\mathcal{N}(0, \sigma^2)$ with zero mean and standard deviation (STD) σ. The network state is updated according to:

$$x_i(t+1) = (\psi_m \circ g)\left(\sum_{j=1}^{N} w_{ij} x_j(t) + u(t)\right),$$

where $g = \tanh$ is the usual hyperbolic tangent nonlinearity and u denotes the input common to all units. At every time step t, the input $u(t)$ is drawn uniformly from $\{-1, 1\}$. The function $\psi_m(\cdot)$ is called quantization function for m bits as it maps from $(-1, 1)$ to its discrete range $\mathcal{S}_m$ of cardinality 2^m:

$$\psi_m : (-1, 1) \to \mathcal{S}_m, \quad \psi_m(x) := \frac{2\lfloor 2^{m-1}(x+1) \rfloor + 1}{2^m} - 1.$$

Here $\lfloor x \rfloor$ denotes the integer part of x. Due to ψ_m the variables $x_i(t)$ are discrete ("quantized") and assume values in $\mathcal{S}_m = \{(2k+1)/2^m - 1 | k = 0, \ldots, 2^m - 1\} \subset (-1, 1)$. The network defined above was utilized for online computations on the input stream $u(\cdot)$. We consider in this article tasks where the binary target output at time t depends solely on the n input bits $u(t-\tau-1), \ldots, u(t-\tau-n)$ for a given delay parameter $\tau \geq 0$, i.e., it is given by $f_T(u(t-\tau-1), \ldots, u(t-\tau-n))$ for a function $f_T \in \{f | f : \{-1, 1\}^n \to \{-1, 1\}\}$. In order to approximate the target output, a linear classifier of the form $\mathrm{sign}(\sum_{i=1}^{N} \alpha_i x_i(t) + b)$ is applied to the instantaneous network state $\mathbf{x}(t)$. The coefficients α_i and the bias b were trained via a one-shot pseudo-inverse regression method (Jaeger, 2001). The RC system consisting of the network and the linear classifier is called a quantized ESN of quantization level m in the remainder of this chapter.

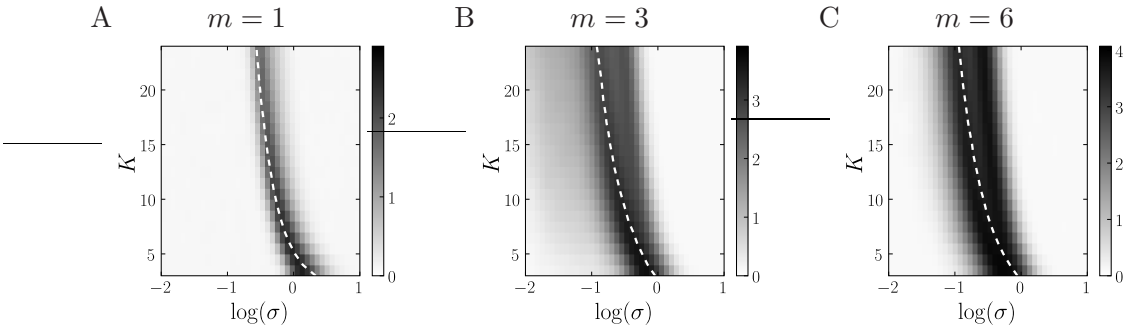


Figure 5.1: The performance $p_{\text{exp}}(C, \text{PAR}_5)$ for three different quantization levels $m = 1, 3, 6$ is plotted as a function of the network in-degree K and the weight STD σ. The networks size is $N = 150$, the results have been averaged over 10 circuits C, initial conditions and randomly drawn input time series of length 10^4 time steps. The dashed line represents the numerically determined critical line.

We assessed the computational capabilities of a given network based on the numerically determined performance on an example task, which was chosen to be the τ-delayed parity function of n bits $\text{PAR}_{n,\tau}$, i.e. the desired output at time t is $\text{PAR}_{n,\tau}(u, t) = \prod_{i=1}^{n} u(t - \tau - i)$ for a delay $\tau \geq 0$ and $n \geq 1$. A separate readout classifier is trained for each combination of n and τ, all using the same reservoir. We define p_{exp} quantifying the performance of a given circuit C on the PAR_n task as:

$$p_{\text{exp}}(C, \text{PAR}_n) := \sum_{\tau=0}^{\infty} \kappa(C, \text{PAR}_{n,\tau}), \tag{5.1}$$

where $\kappa(C, \text{PAR}_{n,\tau})$ denotes the performance of circuit C on the $\text{PAR}_{n,\tau}$ task measured in terms of Cohen's kappa coefficient[2]. The performance results for PAR_n can be considered representative for the general computational capabilities of a circuit C as qualitatively very similar results were obtained for the AND_n task of n bits and random Boolean functions of n bit (results not shown).

In Fig. 5.1 the performance $p_{\text{exp}}(C, \text{PAR}_5)$ is shown averaged over 10 circuits C for three different quantization levels $m = 1, 3, 6$. $p_{\text{exp}}(C, \text{PAR}_5)$ is plotted as a function of the network in-degree K and the logarithm[3] of the weight STD σ. Qualitatively very similar results were obtained for different network graphs with e.g. Poisson or scale-free distributed in-degree with average K (results not shown). A numerical approximation of the critical line, i.e. the order-chaos phase transition, is also shown (dashed line), which was determined by the root of an estimation of the Lyapunov coefficient[4]. The critical

[2] κ is defined as $(c - c_l)/(1 - c_l)$ where c is the fraction of correct trials and c_l is the chance level. The sum in eq. (5.1) was truncated at $\tau = 8$, as the performance was negligible for higher delays $\tau > 8$ for the network size $N = 150$.

[3] All logarithms are taken to the basis 10, i.e. $\log = \log_{10}$ if not stated otherwise.

[4] The Lyapunov coefficient λ was determined in the following way. After 20 initial simulation steps the smallest admissible (for m) state difference $\delta_0(m) = 2^{1-m}$ was introduced in a single network unit and the resulting state difference δ after one time step was measured averaged over 10^5 trials with

line predicts the zone of optimal performance well for $m = 1$, but is less accurate for ESNs with $m = 3, 6$. One can see that for ESNs with low quantization levels ($m = 1, 3$), networks with a small in-degree K reach a significantly better peak performance than those with high in-degree. The effect disappears for a high quantization level ($m = 6$). This phenomenon is consistent with the observation that network connectivity structure is in general an important issue if the reservoir is composed of binary or spiking neurons but less important if analog neurons are employed. Note that for $m = 3, 6$ we see a bifurcation in the zones of optimal performance which is not observed for the limiting cases of ESNs and LSMs.

5.3 Phase Transitions in Binary and High Resolution Networks

Where does the difference between binary and high resolution reservoirs shown in Fig. 5.1 originate from? It was often hypothesized that high computational power in recurrent networks is located in a parameter regime near the critical line, i.e., near the phase transition between ordered and chaotic behavior (see, e.g., (Legenstein & Maass, 2007b) for a review; compare also the performance with the critical line in Fig. 5.1). Starting from this hypothesis, we investigated whether the network dynamics of binary networks near this transition differs qualitatively from the one of high resolution networks. We estimated the network properties by empirically measuring the Lyapunov exponent λ with the same procedure as in the estimation of the critical line in Fig. 5.1 (see text above). However, we did not only determine the critical line (i.e., the parameter values where the estimated Lyapunov exponent crosses zero), but also considered its values nearby. For a given in-degree K, λ can then be plotted as a function of the STD of weights σ (centered at the critical value σ_0 of the STD for that K). This was done for binary (Fig. 5.2A) and high resolution networks (Fig. 5.2B) and for $K = 3, 12$, and 24. Interestingly, the dependence of λ on the STD σ near the critical line is qualitatively quite different between the two types of networks. For binary networks the transition becomes much sharper with increasing K which is not the case for high resolution networks. How can this sharp transition explain the reduced computational performance of binary ESNs with high in-degree K? The tasks considered in this article require some limited amount of memory which has to be provided by the reservoir. Hence, the network dynamics has to be located in a regime where memory about recent inputs is available and past input bits do not interfere with that memory. Intuitively, an effect of the sharper phase transition could be stated in the following way. For low σ (i.e., in the ordered regime), the memory needed for the task is not provided by the reservoir. As we increase σ, the memory capacity increases, but older memories interfere with recent ones, making it hard or even impossible to extract the relevant information. This intuition is confirmed by an analysis which was introduced in (Legenstein & Maass, 2007a) and which we applied to our setup. We estimated two measures of the reservoir, the so

randomly generated networks, initial states and input streams. The initial states of all neurons were iid. uniformly over $\mathcal{S}_m$. λ was then determined by $\lambda = \ln(\delta/\delta_0(m))$.

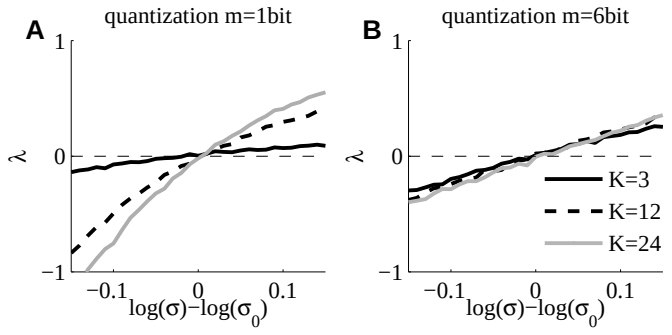


Figure 5.2: Phase transitions in binary networks ($m = 1$) differ from phase transition in high resolution networks ($m = 6$). An empirical estimate λ of the Lyapunov exponent is plotted as a function of the STD of weights σ for in-degrees $K = 3$ (solid), $K = 12$ (dashed), and $K = 24$ (gray line). In order to facilitate comparison, the plot for each K is centered around $\log(\sigma_0)$ where σ_0 is the STD of weights for which λ is zero (i.e., σ_0 is the estimated critical σ value for that K). The transition sharpens with increasing K for binary reservoirs A, whereas it is virtually independent of K for high resolution reservoirs B.

called "kernel-quality" and the "generalization rank", both being the rank of a matrix consisting of certain state vectors of the reservoir. To evaluate the kernel-quality of the reservoir, we randomly drew $N = 150$ input streams $u_1(\cdot), \ldots, u_N(\cdot)$ and computed the rank of the $N \times N$ matrix whose columns were the circuit states resulting from these input streams.[5] Intuitively, this rank measures how well the reservoir represents different input streams. The generalization rank is related to the ability of the reservoir-readout system to generalize from the training data to test data. The generalization rank is evaluated as follows. We randomly drew N input streams $\tilde{u}_1(\cdot), \ldots, \tilde{u}_N(\cdot)$ such that the last three input bits in all these input streams were identical.[6] The generalization rank is then given by the rank of the $N \times N$ matrix whose columns are the circuit states resulting from these input streams. Intuitively, the generalization rank with this input distribution measures how strongly the reservoir state at time t is sensitive to inputs older than three time steps. The rank measures calculated here will thus have predictive power for computations which require memory of the last three time steps (see (Legenstein & Maass, 2007a) for a theoretical justification of the measures). In general, a high kernel-quality and a low generalization rank (corresponding to a high ability of the network to generalize) are desirable. Fig. 5.3A and D show the difference between the two measures as a function of $\log(\sigma)$ and the in-degree K for binary networks and high resolution networks respectively. The plots show that the peak value of this difference is decreasing with K in binary networks, whereas it is independent of K in high resolution reservoirs, reproducing the observations in the plots for the computational

[5] The initial states of all neurons were iid. uniformly over $\mathcal{S}_m$. The rank of the matrix was estimated by singular value decomposition on the network states after 15 time steps of simulation.

[6] First, we drew each of the last three bits $\tilde{u}(13), \ldots, \tilde{u}(15)$ independently from a uniform distribution over $\{-1, 1\}$. For each input stream $\tilde{u}_i(1), \ldots, \tilde{u}_i(15)$ we drew $\tilde{u}_i(1), \ldots, \tilde{u}_i(12)$ independently from a uniform distribution over $\{-1, 1\}$ and set $\tilde{u}_i(t) = \tilde{u}(t)$ for $t = 13, \ldots, 15$.

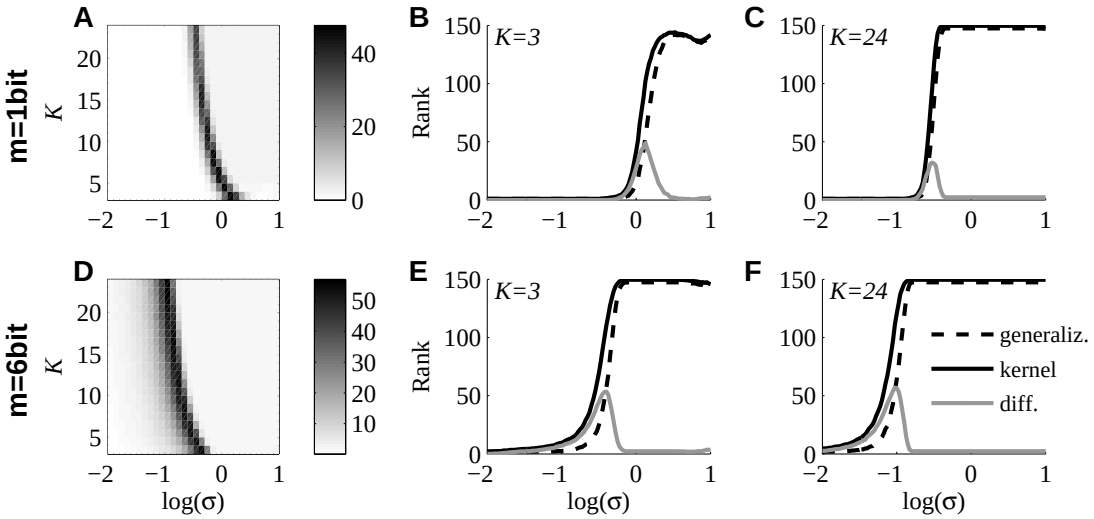


Figure 5.3: Kernel-quality and generalization rank of quantized ESNs of size $N = 150$. Upper plots are for binary reservoirs ($m = 1$bit), lower plots for high resolution reservoirs ($m = 6$ bit). A: The difference between the kernel-quality and the generalization rank as a function of the log STD of weights and the in-degree K. B: The kernel-quality (solid), the generalization rank (dashed) and the difference between both (gray line) for $K = 3$ as a function of $\log(\sigma)$. C: Same as panel B, but for an in-degree of $K = 24$. In comparison to panel B, the transition of both measures is much steeper. D,E,F: Same as panels A, B, and C respectively, but for a high resolution reservoir. All plotted values are means over 100 independent runs with randomly drawn networks, initial states, and input streams.

performance. A closer look for the binary circuit at $K = 3$ and $K = 24$ is given in Fig. 5.3B and 5.3C. When comparing these plots, one sees that the transition of both measures is much steeper for $K = 24$ than for $K = 3$ which leads to a smaller difference between the measures. We interpret this finding in the following way. For $K = 24$, the reservoir increases its separation power very fast as $\log(\sigma)$ increases. However the separation of past input differences increases likewise and thus early input differences cannot be distinguished from late ones. This reduces the computational power of binary ESN with large K on such tasks. In comparison, the corresponding plots for high resolution reservoirs (Figs. 5.3E and 5.3F) show that the transition shifts to lower weight STDs σ for larger K, but apart from this fact the transitions are virtually identical for low and high K values. Comparing Fig. 5.3D with Fig. 5.1C, one sees that the rank measure does not accurately predict the whole region of good performance for high resolution reservoirs. It also does not predict the observed bifurcation in the zones of optimal performance, a phenomenon that is reproduced by the mean-field predictor introduced in the following section.

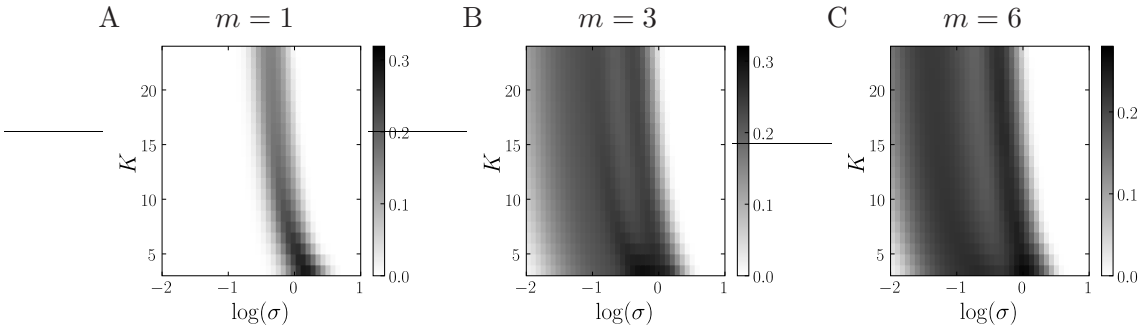


Figure 5.4: Mean-field predictor p_∞ for computational power for different quantization levels m as a function of the STD σ of the weights and in-degree K. A: $m = 1$. B: $m = 3$. C: $m = 6$. Compare this result to the numerically determined performance p_{exp} plotted in Fig. 5.1.

5.4 Mean-Field Predictor for Computational Performance

The question why and to what degree certain non-autonomous dynamical systems are useful devices for online computations has been addressed theoretically amongst others in (Bertschinger & Natschlaeger, 2004). There, the computational performance of networks of randomly connected threshold gates was linked to their separation property (for a formal definition see (Maass et al., 2002)): It was shown that only networks which exhibit sufficiently different network states for different instances of the input stream, i.e. networks that separate the input, can compute complex functions of the input stream. Furthermore, the authors introduced an accurate predictor for the computational capabilities for the considered type of networks based on the separation capability which was quantified via a simple mean-field approximation of the Hamming distance between different network states.

Here we aim at extending this approach to a larger class of networks, the class of quantized ESNs introduced above. However a severe problem arises when directly applying the mean-field theory developed in (Bertschinger & Natschlaeger, 2004) to quantized ESNs with a quantization level $m > 1$: Calculation of the important quantities becomes computationally infeasible as the state space of a network grows exponentially with m. Therefore we introduce a modified mean-field predictor which can be efficiently computed and which still has all desirable properties of the one introduced in (Bertschinger & Natschlaeger, 2004).

Suppose the target output of the network at time t is a function $f_T \in F = \{f | f : \{-1, 1\}^n \to \{-1, 1\}\}$ of the n bits $u(t - \tau - 1), \ldots, u(t - \tau - n)$ of the input stream $u(\cdot)$ with delay τ as described in Sec. 5.2. In order to exhibit good performance on an arbitrary $f_T \in F$, pairs of inputs that differ in at least one of the n bits have to be mapped by the network to different states at time t. Only then, the linear classifier is able to assign the inputs to different function values. In order to quantify this so-called separation property of a given network, we introduce the normalized distance $d(k)$: It measures the average distance between two networks states

$\mathbf{x}^1(t) = (x_1^1(t), \ldots, x_N^1(t))$ and $\mathbf{x}^2(t) = (x_1^2(t), \ldots, x_N^2(t))$ arising from applying to the same network two input streams $u^1(\cdot)$ and $u^2(\cdot)$ which only differ in the single bit at time $t-k$, i.e. $u^2(t-k) = -u^1(t-k)$. Formally we define[7]:

$$d(k) = \frac{1}{N} \left\langle \left\| \mathbf{x}^1(t) - \mathbf{x}^2(t) \right\|_1 \right\rangle .$$

The average $\langle . \rangle$ is taken over all inputs $u^1(\cdot)$, $u^2(\cdot)$ from the ensemble defined above, all initial conditions of the network and all circuits C. However, a good separation of the n bits, i.e. $d(k) \gg 0,\ \tau < k \leq n+\tau$, is a necessary but not a sufficient condition for the ability of the network to calculate the target function. Beyond this, it is desired that the network "forgets" all (for the target function) irrelevant bits $u(t-k),\ k > n+\tau$ of the input sufficiently fast, i.e. $d(k) \approx 0$ for $k > n+\tau$. We use the limit $d(\infty) = \lim_{k\to\infty} d(k)$ to quantify this irrelevant separation which signifies sensitivity to initial conditions (making the reservoir not time invariant). Hence, we propose the quantity p_∞ as a heuristic predictor for computational power:

$$p_\infty = \max \{d(2) - d(\infty), 0\} .$$

As the first contribution to p_∞ we chose $d(2)$ as it reflects the ability of a network to perform a combination of two mechanisms: In order to exhibit a high value for $d(2)$ the network has to separate the inputs at the time step $t-2$ and to sustain the resulting state distance via its recurrent dynamics in the next time step $t-1$. We therefore consider $d(2)$ to be a measure for input separation on short time-scales relevant for the target function. p_∞ is calculated using a mean-field model similar to the one presented in (Bertschinger & Natschlaeger, 2004) which itself is rooted in the annealed approximation (AA) introduced in (Derrida & Pomeau, 1986). In the AA one assumes that the circuit connectivity and the corresponding weights are drawn iid. at every time step. Although being a drastic simplification, the AA has been shown to yield good results in the large system size limit $N \to \infty$. The main advantage of p_∞ over the the predictor defined in (Bertschinger & Natschlaeger, 2004) (the NM-separation) is that the calculation of p_∞ only involves taking the average over one input stream (as the $u^2(\cdot)$ is a function of $u^1(\cdot)$) compared to taking the average over two independent inputs needed for the NM-separation, resulting in a significantly reduced computation time.

In Fig. 5.4 the predictor p_∞ is plotted as a function of the STD σ of the weight distribution and the in-degree K for three different values of the quantization level $m \in \{1, 3, 6\}$. When comparing these results with the actual network performance $p_{\text{exp}}(\text{PAR})$ on the PAR-task plotted in Fig. 5.1 one can see that p_∞ serves as a reliable predictor for p_{exp} of a network for sufficiently small m. For larger values of m the predictor p_∞ starts to deviate from the true performance. The dominant effect of the quantization level m on the performance discussed in Sec. 5.2 is well reproduced by p_∞: For $m = 1$ the in-degree K has a considerable impact, i.e. for large K maximum performance drops significantly. For $m > 2$ however, for larger values of K there also exists a region in the parameter space exhibiting maximum performance.

[7]For vectors $\mathbf{x} = (x_1, x_2, \ldots) \in \mathbb{R}^N$ we use the Manhattan norm $\|\mathbf{x}\|_1 := \sum_{i=1}^N |x_i|$

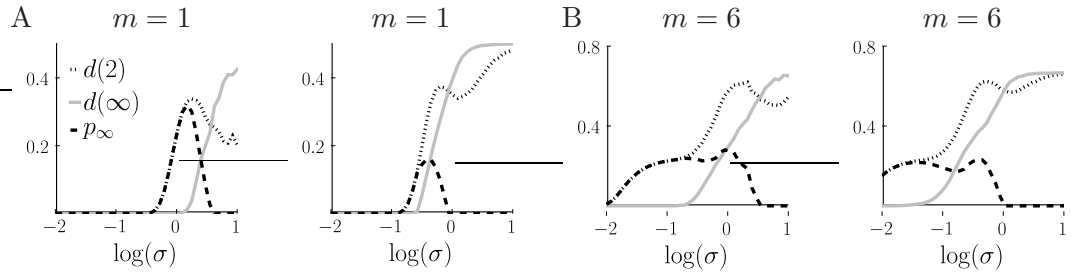


Figure 5.5: Contributions $d(2)$ (dotted) and $d(\infty)$ (solid gray) to the mean-field predictor p_∞ (dashed line) for different quantization levels $m \in \{1, 6\}$ and different in-degrees $K \in \{3, 24\}$ as a function of STD σ of the weights. The plots show slices of the 2d plots Fig. 5.4A and C for constant K. A: For $m = 1$ it can be seen that the region in $\log(\sigma)$-space with high $d(2)$ and low $d(\infty)$ is significantly larger for $K = 3$ than for $K = 24$. B: For $m = 6$ this region is roughly independent of K except a shift.

The interplay between the two contributions $d(2)$ and $d(\infty)$ of p_∞ delivers insight into the dependence of p_{exp} on the network parameters. A high value of $d(2)$ corresponds to a good separation of inputs on short time scales relevant for the target task, a property that is found predominantly in networks that are not strongly input driven. A small value of $d(\infty)$ guarantees that inputs on which the target function assumes the same value are mapped to nearby network states and thus a linear readout is able to assign them to the same class irrespectively of their irrelevant remote history. For $m = 1$, as can be seen in Fig. 5.5 the region in $\log(\sigma)$ space where both conditions for good performance are present decreases for growing K. In contrast, for $m > 2$ a reverse effect is observed: for increasing K the parameter range for σ fulfilling the two opposing conditions for good performance grows moderately resulting in a large region of high p_∞ for high in-degree K. This observation is in close analogy to the behavior of the rank measure discussed in Sec. 5.3. Also note that p_∞ predicts the novel bifurcation effect also observed in Fig. 5.1.

5.5 Discussion

By interpolating between the ESN and LSM approaches to RC, this work provides new insights into the question of what properties of a dynamical system lead to improved computational performance: Performance is optimal at the order-chaos phase transition, and the broader this transition regime, the better will the performance of the system be. We have confirmed this hypothesis by several analyses, including a new theoretical mean-field predictor that can be computed very efficiently.The importance of a gradual order-chaos phase transition could explain why ESNs are more often used for applications than LSMs. Although they can have very similar performance on a given task (Verstraeten et al., 2007), it is significantly harder to create a LSM which operates at the edge-of-chaos: the excitation and inhibition in the network need to be finely balanced because there tends to be a very abrupt transition from an ordered to a epileptic state. For ESNs

however, there is a broad parameter range in which they perform well. It should be noted that the effect of quantization cannot just be emulated by additive or multiplicative iid. or correlated Gaussian noise on the output of analog neurons. The noise degrades performance homogeneously and the differences in the influence of the in-degree observed for varying quantization levels cannot be reproduced. The finding that binary reservoirs have superior performance for low in-degree stands in stark contrast to the fact that cortical neurons have very high in-degrees of over 10^4. This raises the interesting question which properties and mechanisms of cortical circuits not accounted for in this article contribute to their computational power. In view of the results presented in this article, such mechanisms should tend to soften the phase transition between order and chaos.

5.5.0.1 Acknowledgments

This chapter is based on the paper *On computational power and the order-chaos phase transition in reservoir computing*, which was written by Benjamin Schrauwen (BS), Lars Büsing (LB) and Robert Legenstein (RL). BS investigated the differences between binary and analog reservoirs and designed the simulations. LB developed the newtork model, performed the simulations and developed the mean-field predictor. RL contributed the rank measure analysis. LB, BS and RL wrote the manuscript.

Chapter 6

Connectivity, Dynamics and Memory in Reservoir Computing with Binary and Analog Neurons

Contents

Reservoir Computing (RC) systems are powerful models for online computations on input sequences. They consist of a memoryless readout neuron which is trained on top of a randomly connected recurrent neural network. RC systems are commonly used in two flavors: with analog or binary (spiking) neurons in the recurrent circuits. Previous work indicated a fundamental difference in the behavior of these two implementations of the RC idea. The performance of a RC system built from binary neurons seems to depend strongly on the network connectivity structure. In networks of analog neurons such clear dependency has not been observed. In this article we address this apparent dichotomy by investigating the influence of the network connectivity (parametrized by the neuron in-degree) on a family of network models that interpolates between analog and binary networks. Our analyses are based on a novel estimation of the Lyapunov exponent of the network dynamics with the help of branching process theory, rank measures which estimate the kernel-quality and generalization capabilities of recurrent networks, and a novel mean-field predictor for computational performance. These analyses reveal that the phase transition between ordered and chaotic network behavior of binary circuits qualitatively differs from the one in analog circuits, leading to differences in the integration of information over short and long time scales. This explains the decreased computational performance observed in binary circuits that are densely connected. The

mean-field predictor is also used to bound the memory function of recurrent circuits of binary neurons.

6.1 Introduction

The idea of using a randomly connected recurrent neural network for online computations on an input sequence was independently introduced in (Jaeger, 2001) and (Maass et al., 2002). In these papers the network activity is regarded as an "echo" of the recent inputs and a memoryless readout device is then trained in order to approximate from this "echo" a given time-invariant target operator with fading memory (see (Maass et al., 2002)) whereas the network itself remains untrained. Jaeger used analog sigmoidal neurons as network units and named the model Echo State Network (ESN). Maass termed the idea Liquid State Machine (LSM) and most of the related literature focuses on networks of spiking neurons or threshold units. Both ESNs and LSMs are special implementations of a concept now generally called Reservoir Computing (RC) which subsumes the idea of using general dynamical systems (e. g. a network of interacting optical amplifiers (Vandoorne et al., 2008), or an analog VLSI Cellular Neural Network chip (Verstraeten et al., 2008)) – the so-called reservoirs – in conjunction with trained memoryless readout functions as computational devices. These RC systems have been used in a broad range of applications (often outperforming other state-of-the-art methods) such as chaotic time-series prediction and non-linear wireless channel equalization (Jaeger & Haas, 2004), speech recognition (Verstraeten et al., 2005; Jaeger, Lukosecicius, Popovici, & Siewert, 2007), movement analysis (Legenstein, Markram, & Maass, 2003), and robot control (Joshi & Maass, 2005).

Although ESNs and LSMs are based on very similar ideas (and in applications it seems possible to switch between both approaches without loss of performance (Verstraeten et al., 2007)) an apparent dichotomy exists regarding the influence of the reservoir's connectivity on its computational performance. The performance of an ESN using analog, rate-based neurons is e. g. largely independent of the sparsity of the network (Jaeger, 2007) or the exact network topology such as small-world or scale-free connectivity graphs[1]. For LSMs, which consist of spiking or binary units, a profoundly different effect has been observed, e. g. introducing small-world or biologically measured lamina-specific cortical interconnection statistics (Haeusler & Maass, 2007) clearly leads to an increase in performance. Further, in the results of (Bertschinger & Natschlaeger, 2004) it can be observed (although not specifically stated there) that for networks of threshold gates with a simple connectivity topology of fixed in-degree per neuron, an increase in performance can be found for decreasing in-degree. None of these effects can be reproduced using ESNs.

In order to systematically study this fundamental difference between binary (spik-

[1]Shown by results of unpublished experiments which have also been reported by the lab of Jaeger through personal communication. Of course, drastic topological changes such as disconnecting all network units influences performance. Further, a specific class of (nonnormal) connectivity matrices with advantageous memory properties for linear systems was characterized in (Ganguli, Huh, & Sompolinsky, 2008).

ing) LSMs and analog ESNs in a unified framework, we close the gap between these models by introducing in Section 6.2 a class of models termed quantized ESNs (qESNs). The reservoir of a quantized ESN is defined as a network of discrete-valued units, where the number of admissible states of a single unit is controlled by a parameter called state resolution which is measured in bits. A binary network (where the units have binary outputs) has a state resolution of 1, whereas high resolution networks (where the units provide high resolution outputs) have high state resolutions. LSMs and ESNs can thus be interpreted as the two limiting cases of quantized ESNs for low and high state resolution respectively. We briefly described the dynamics of qESNs, which exhibit ordered and chaotic behavior separated by a phase transition. Further the concept of Lyapunov exponents is discussed in the context of qESNs and an approach to approximately compute them for qESNs is introduced based on branching process theory.

In Section 6.3 we numerically study the influence of the network connectivity parametrized by the in-degree of the network units on the computational performance of quantized ESNs for different state resolutions. This generalizes and systemizes previous results obtained for binary LSMs and analog ESNs.

In Section 6.4 the empirical results are analyzed by studying the Lyapunov exponent of qESNs, which exhibits a clear relation to the computational performance (Legenstein & Maass, 2007a). We show that for binary qESNs, the chaos-order phase transition is significantly more gradual when the networks are sparsely connected. It is exactly in this transition regime that the computational power of a Reservoir Computing system is found to be optimal (Legenstein & Maass, 2007a). This effect disappears for high-resolution ESNs.

A clear explanation of the influence of the network connectivity on the computational performance can be found by investigating the rank measure presented in (Legenstein & Maass, 2007a). This measure characterizes the computational capabilities of a network as a trade-off between the so-called kernel-quality and the generalization ability. We show that for highly connected binary reservoirs the region of an efficient trade-off implying high performance is narrow. For sparser networks this region is shown to broaden. Consistently for high resolution networks the region is found to be independent of the interconnection degree.

In Section 6.5 we present a novel mean-field predictor for computational power which is able to reproduce the influence of the connectivity on the qESN model. This predictor is based on an estimation of the input separation property of a network and it is a generalization of the predictor introduced in (Bertschinger & Natschlaeger, 2004) as it can be calculated for general (not just binary) qESNs and it can be determined with a significantly reduced computation time. This enables us to study computational power based on state separation for recurrent networks of nearly analog nonlinear neurons. The novel theoretical measure matches the experimental and rank measure findings closely. It describes high performance as an interplay between input separation on different time-scales revealing important properties of RC systems necessary for good computational capabilities.

We investigate the relation of the mean-field predictor to the memory function of qESNs in Section 6.6 and show that the latter is bounded by a function of the input

separation property. To our best knowledge this represents the first computationally feasible bound on the memory function of nonlinear networks, since such bounds were previously only derived for linear networks (White, Lee, & Sompolinsky, 2004; Jaeger, 2002; Ganguli et al., 2008). Further, we investigate the scaling of the memory capacity and the temporal capacity of binary qESNs with the network size yielding a logarithmic dependence.

The numerical and theoretical finding that for binary qESNs the optimal performance is observed for sparse connectivity is compared to experimental results regarding the connectivity of neocortical microcircuits in Section 6.7 revealing an apparent discrepancy between the optimal parameter values of the binary qESN model and experimental observations. Sparse network activity which is ubiquitous in biological spiking networks but absent in the qESN model is proposed as a possible mechanism that can resolve this discrepancy.

6.2 Quantized ESNs and Their Dynamics

In this section the network model is defined that will later serve as the dynamic reservoir for qESNs. The networks are reminiscent of random Boolean networks (see (Shmulevich, Dougherty, Kim, & Zhang, 2002)) and Kauffman networks (see (Kauffman, 1969; Derrida & Stauffer, 1986)) and exhibit just like the latter two distinct dynamical regimes, the chaotic and the ordered regime, depending on the choice of parameters. These two "phases" are separated by a order-chaos (also termed order-disorder) phase transition (for general information on phase transitions see (Zinn-Justin, 2003)).

Definition of the Network Model

We consider networks of N units with the state variable $\mathbf{x}(t) = (x_1(t), \ldots, x_N(t)) \in [-1, +1]^N$ in discrete time $t \in \mathbb{Z}$. All units have an in-degree of K, i.e. every unit i receives input from K other randomly chosen units with independently identically distributed (iid.) weights w_{ij} drawn from a normal distribution $\mathcal{N}(0, \sigma^2)$ with zero mean and standard deviation (STD) σ. All remaining weights are defined as zero. Sample weight matrices $(w_{ij})_{i,j\in\{1,\ldots,N\}}$ (constituting concrete networks/circuits) from this distribution will be denoted by C in the remainder. The network state is updated according to:

$$x_i(t+1) = (\psi_m \circ g)\left(\sum_{j=1}^{N} w_{ij} x_j(t) + u(t)\right), \tag{6.1}$$

where $g = \tanh$ is the usual hyperbolic tangent nonlinearity and u denotes the input sequence common to all units. At every time step t, the input $u(t)$ is drawn uniformly from $\{-1, 1\}$. The function ψ_m is called quantization function for m bits as it maps from $(-1, 1)$ to its discrete range $\mathcal{S}_m$ of cardinality 2^m:

$$\psi_m : (-1, 1) \to \mathcal{S}_m, \quad \psi_m(x) := \frac{2\lfloor 2^{m-1}(x+1)\rfloor + 1}{2^m} - 1.$$

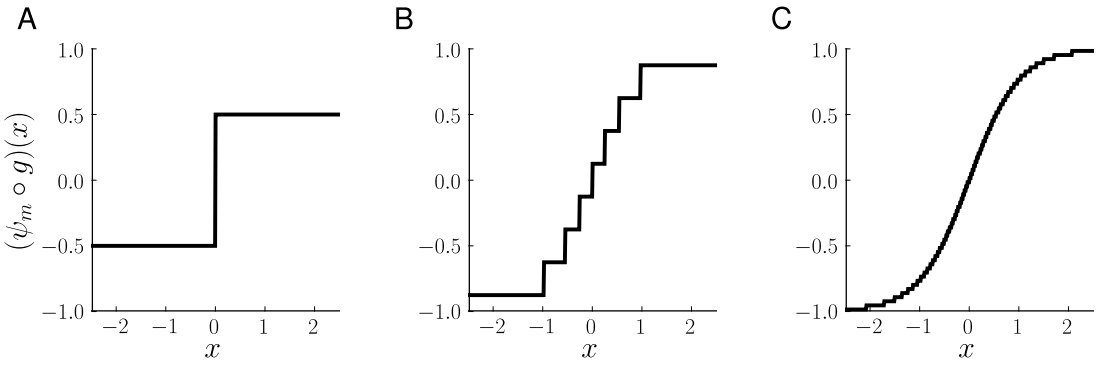


Figure 6.1: The quantized activation function $\psi_m \circ g$ for state resolutions $m = 1$ (panel A), $m = 3$ (panel B) and $m = 6$ (panel C). Networks with the $m = 1$ activation function, which just assumes two different values, are termed binary reservoirs whereas networks with large m (here $m = 6$) are termed high-resolution reservoirs as their units posses a large state space of 2^m states. The discretization schema was was chosen such that states are equidistant and $\sum_{i=1}^{2^m} s_i p(s_i) = 0$ as well as $\sum_{i=1}^{2^m} |s_i| p(s_i) = 1/2$ independent of the state resolution m assuming a uniform distribution over the single unit state space $p(s_i) = 2^{-m}$.

Here $\lfloor x \rfloor$ denotes the integer part of x. Due to ψ_m the variables $x_i(t)$ are discrete ("quantized") and assume values in the state space $\mathcal{S}_m = \{s_1, \ldots, s_{2^m}\} \subset (-1, 1)$ with $s_k := (2k-1)/2^m - 1$. Three examples of the quantized activation function $\psi_m \circ g$ for state resolutions $m = 1, 3, 6$ are shown in Fig. 6.1.

Depending on the parameters m, K, and σ the system defined by (6.1) shows two qualitatively different behaviors namely ordered and chaotic dynamics which are separated in the parameter space by a sharp boundary, often called the critical line, where a phase transition takes place. The ordered (also called "frozen") and chaotic regimes are defined via the average damage spreading properties (i. e. sensitivity to perturbations of initial conditions) of the networks (Derrida & Pomeau, 1986; Derrida & Stauffer, 1986; Bertschinger & Natschlaeger, 2004). One considers the temporal evolution of the average distance $H(t) = \left\langle \|\mathbf{x}^1(t) - \mathbf{x}^2(t)\| \right\rangle_{C,u}$ between two states $\mathbf{x}^1(t)$ and $\mathbf{x}^2(t)$ at time t evolving from initial conditions that differ in a single unit at time 0. Here $\|\cdot\|$ denotes some norm in $\mathbb{R}^N$ e. g. the p-1 norm $\|\cdot\|_1$ and $\langle\cdot\rangle_{C,u}$ denotes the average over networks C (with the same parameters m, σ, K), initial perturbations and input sequences u. A set of parameters m, K, σ is in the ordered phase if $\lim_{t\to\infty} H(t) =: H^* = 0$, i. e. if perturbations in the initial conditions $H(0)$ eventually die out. Parameter sets with $H^* > 0$ are in the chaotic regime where the initial "damage" $H(0)$ persists and influences the network state for all later times. Hence H^* can be considered an order parameter of the phase transition as it is zero in one phase and larger than zero in the other. Examples for the behavior of H^* for state resolutions $m = 1, 3, 6$ and varying parameters σ and K are shown in Fig. 6.2 exhibiting the characteristic behavior of a phase transition.

The state distance $H(t)$ also allows the introduction of a heuristic measure for the speed of divergence of trajectories in discrete-time and discrete state-space systems.

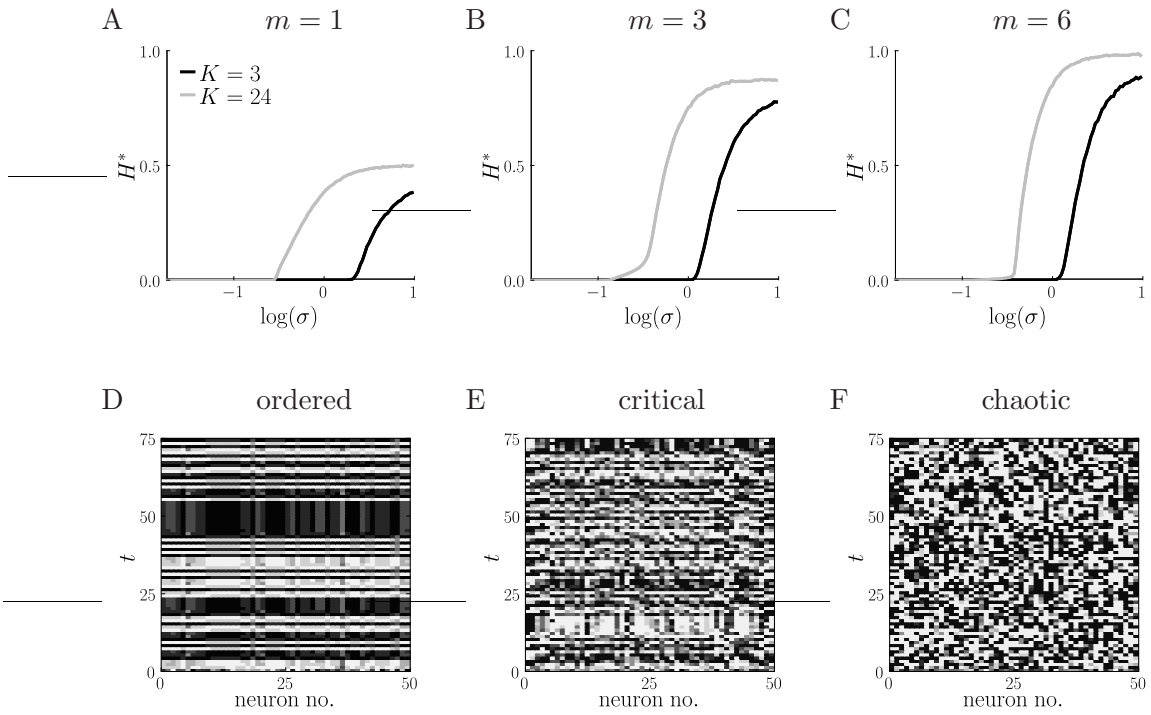


Figure 6.2: Phase transitions in randomly connected networks with dynamics defined by (6.1). A, B, C: Shown is the fixed point H^*/N of the normalized distance $H(t)/N$ between two states evolving from different initial conditions in networks with $N = 500$ units for three state resolutions $m = 1, 3, 6$ and varying in-degree K and weight standard deviation σ. The abrupt change in the values of H^* for different parameters is characteristic for a phase transition. Shown results are averages over 500 circuits (with a single random input sequence each). The initial perturbation $H(0)$ was chosen as the smallest admissible perturbation (for the specific m) in a single unit and H^* was measured after 100 update steps. D, E, F: Evolution of the state $x_i(t)$ of 75 out of $N = 500$ units from a network with $m = 3$ and $K = 3$ for $\log(\sigma) = -0.5$ (panel D), $\log(\sigma) = 0$ (panel E) and $\log(\sigma) = 0.5$ (panel F) showing ordered, critical and chaotic dynamics respectively.

We will term this measure Lyapunov exponent λ as it is reminiscent of the maximal Lyapunov exponent in systems with continuous (see (Katok & Hasselblatt, 1995)) and binary state space (see (Luque & Solé, 2000)). It is defined via:

$$\lambda = \lim_{T\to\infty} \frac{1}{T} \ln\left(\frac{H(T)}{H(0)}\right). \tag{6.2}$$

In the ordered regime we have $\lambda < 0$ while $\lambda > 0$ holds for chaotic dynamics. The above definition of λ only makes sense for infinitely large systems.

In finite size systems we characterize the speed of divergence of nearby trajectories by a numerical estimation λ_{exp} of the the Lyapunov exponent that was determined in the following way. After 20 initial simulation steps the smallest admissible (for m) state difference $\delta_0(m) = 2^{1-m}$ was introduced in a single network unit and the resulting state difference δ after one time step was measured averaged over 10^5 trials with randomly generated networks and initial states. The initial states of all neurons were iid. uniformly over $\mathcal{S}_m$. The estimation λ_{exp} was then determined by $\lambda_{\text{exp}} = \ln(\delta/\delta_0(m))$. This one step approximation of λ is expected to produce accurate results for large networks with sparse connectivity where all "damages" spread independently through the network. It is shown below that already at a networks size of $N = 150$, λ and λ_{exp} agree well for a large regime of network parameters.

Lyapunov Exponents via Branching Processes

Besides estimating the Lyapunov exponent defined in (6.2) using the scheme for finite size systems described above, it can be calculated for infinitely large systems $N \to \infty$ under the annealed approximation (AA) using results from the theory of multitype branching processes (see (Athreya & Ney, 1972)). In the AA that was introduced in (Derrida & Pomeau, 1986) one assumes that the circuit connectivity and the corresponding weights are drawn iid. at every time step. Although being a drastic simplification, the AA has been shown in various studies (see (Derrida & Pomeau, 1986; Bertschinger & Natschlaeger, 2004; White et al., 2004)) to be a powerful tool for investigating network dynamics yielding accurate results for large system sizes N, hence its application is well justified in the limit $N \to \infty$ considered here. Branching process theory has already been applied in theoretical neuroscience to describe the temporal/spacial dynamics of neural activity (see (Beggs & Plenz, 2003; Vogels, Rajan, & Abbott, 2005)). Here we propose the novel approach to apply branching process theory for studying the evolution of perturbations of networks states allowing the approximate evaluation of the Lyapunov spectrum of the network model defined above.

Let $\mathcal{S}_m = \{s_1, \ldots, s_{2^m}\}$ denote the single unit state space with a state resolution m. Consider two states $\mathbf{x}^1(t)$ and $\mathbf{x}^2(t)$ of the same infinitely large network (that e. g. evolved from different initial conditions). We say that there is a perturbation of type $s_i \to s_j$ at time t (of $\mathbf{x}^1(t)$ relative to $\mathbf{x}^2(t)$) if there is a network unit, with some index $l \in \mathbb{N}$, which is in the state s_i in $\mathbf{x}^1(t)$ and which is in the state s_j in $\mathbf{x}^2(t)$, i. e. $x_l^1(t) = s_i$ and $x_l^2(t) = s_j$. Assuming the AA the neuron indices can be permuted arbitrarily (as the weight matrix is regenerated at every time step) and hence the difference between the

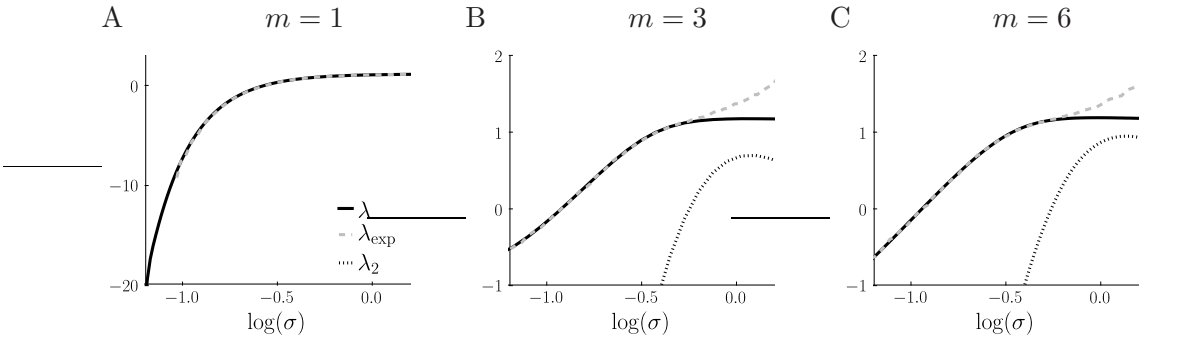


Figure 6.3: The numerically determined Lyapunov exponent $\lambda_{\exp}$ (for $N = 150$), the largest and second largest Lyapunov exponents λ and λ_2 (for $m \neq 1$) obtained by branching process theory are plotted as a function of the weight scale σ for $K = 24$ and three different state resolutions $m = 1, 3, 6$. If $\lambda_2 < 0$, λ and $\lambda_{\exp}$ agree well, whereas $\lambda_{\exp}$ is larger than λ for weight scales σ with $\lambda_2 > 0$. This can be explained by the fact that $\lambda_{\exp}$ measures the total rate of perturbation growth which is governed by all positive Lyapunov exponents hence in particular by λ and λ_2.

two states $\mathbf{x}^1(t)$ and $\mathbf{x}^2(t)$ is fully described by counting the perturbations of all types. Now let $\mathbf{x}^1(t)$ and $\mathbf{x}^2(t)$ differ in n coordinates, which can without loss of generality be assumed to be the first n coordinates, i.e. $x_i^1(t) = s_{a_i} \neq x_i^2(t) = s_{b_i}$ with $a_i,\ b_i \in \{1, \ldots, 2^m\}$ for $i = 1, \ldots, n$ and $x_i^1(t) = x_i^2(t)$ for $i > n$. These n perturbations of types $s_{a_1} \to s_{b_1}, \ldots, s_{a_n} \to s_{b_n}$ at time t cause perturbations in the next time step $t + 1$. Because of the finite in-degree K and the infinite system size $N = \infty$, these n perturbations give rise to "descendant" perturbations at $t+1$ independently. Therefore this system is equivalent to a multitype branching process with $2^m \cdot (2^m - 1)$ types (diagonal perturbations $s_a \to s_a$ do not contribute), a mathematical model which has been extensively studied (see (Athreya & Ney, 1972)). The multitype branching process describing the perturbation spreading in the considered network is fully specified by $p_{i,j}^{\alpha,\beta}$ for $\alpha, \beta, i, j = 1, \ldots, 2^m$ denoting the probability of a $s_\alpha \to s_\beta$ perturbation to cause a $s_i \to s_j$ perturbation per outgoing link in the next time step which can be explicitly calculated using the AA (see appendix F.1). Applying the results from branching theory (see (Athreya & Ney, 1972)) the maximal Lyapunov exponent λ defined in (6.2) is given by the logarithm of the largest eigenvalue (being the Perron root) of the matrix M, whose entries $M_{\alpha+2^m\beta, i+2^m j} = K \cdot p_{i,j}^{\alpha,\beta}$ denote the mean number of descendants of type $s_i \to s_j$ caused by a $s_\alpha \to s_\beta$ perturbation. Branching processes with $\lambda < 0$ are called subcritical, corresponding to ordered dynamics, implying that all perturbations eventually die out with probability one, whereas the case $\lambda > 0$ is termed supercritical, corresponding to chaotic dynamics, implying exponential growth of the number of perturbations on average. For $m > 1$ there is more than one eigenvalue of M giving rise to a Lyapunov spectrum λ_i for $i = 1, \ldots, 2^{m-1}(2^m - 1)$ with $\lambda_i \geq \lambda_{i+1}$ and $\lambda_1 = \lambda$.

In Fig. 6.3 the numerically determined Lyapunov exponent $\lambda_{\exp}$ (for $N = 150$), the largest Lyapunov exponent λ as well as the second largest one λ_2 (for $m \neq 1$) obtained

by branching process theory are plotted as a function of the weight STD σ for $K = 24$ and three different state resolutions $m = 1, 3, 6$. It can be observed that as long as $\lambda_2 < 0$, the branching process exponent λ predicts well the numerically determined exponent λ_{exp}. For $\lambda_2 > 0$, λ_{exp} is larger than λ as in this case λ_2 also contributes to the total growth rate of the number of perturbations which is numerically estimated by λ_{exp}. Hence it can be concluded that the Lyapunov spectrum determined by branching process theory using the AA characterize the perturbation dynamics in the case of finite size systems quite accurately.

6.3 Online Computations with Quantized ESNs

In this section we numerically investigate the capabilities of the networks defined in Section 6.2 for online computations on the binary input sequence u using the RC approach, i. e. the networks are augmented by a trainable readout device, which is in our context a simple linear classifier. In this article we consider tasks where the binary target output at time t depends solely on n input bits in the recent past, i. e. on the n input bits $u(t-\tau-1), \ldots, u(t-\tau-n)$ for given $n \geq 1$ and delay parameter $\tau \geq 0$. More precisely the target output is given by $f_T(u(t-\tau-1), \ldots, u(t-\tau-n))$ for a function $f_T \in \{f | f : \{-1, 1\}^n \to \{-1, 1\}\}$. In order to approximate the target output at time t a linear classifier with the output $\text{sign}(\sum_{i=1}^{N} \alpha_i x_i(t) + b)$ at time t is applied to the instantaneous network state $\mathbf{x}(t)$. The coefficients α_i and the bias b were trained via a pseudo-inverse regression method (see (Jaeger, 2001)). The RC system consisting of a network defined by (6.1) with parameters m, K, σ and a linear classifier is called a quantized ESN (qESN) of state resolution m in the remainder of this chapter.

We assessed the computational capabilities of a given network C based on the numerically determined performance on an example task, which was chosen to be the τ-delayed parity function of n bits $\text{PAR}_{n,\tau}$, i. e. the desired output at time t is $\text{PAR}_{n,\tau}(u, t) = \prod_{i=1}^{n} u(t-\tau-i)$ for a delay $\tau \geq 0$ and $n \geq 1$. A separate readout classifier is trained for each combination of n and τ, all using the same network C as reservoir. We define p_{exp} which quantifies the experimentally determined computational performance of a given circuit C on the PAR_n task as:

$$p_{\text{exp}}(C, \text{PAR}_n) := \sum_{\tau=0}^{\infty} \kappa(C, \text{PAR}_{n,\tau}). \tag{6.3}$$

Here $\kappa(C, \text{PAR}_{n,\tau}) \in [0, 1]$ denotes the performance of circuit C on the $\text{PAR}_{n,\tau}$ task measured in terms of Cohen's kappa coefficient [2] (where 0 corresponds to chance and 1 to optimal performance). To facilitate intuition, in Fig. F.1 of appendix F.2 the dependence of Cohen's kappa coefficient on the delay τ and the network size N is illustrated for two different parameter settings. According to it's definition the performance measure p_{exp} sums up the kappa coefficients for all delays τ. For example, a network C which

[2] κ is defined as $(c - c_l)/(1 - c_l)$ where c is the fraction of correct trials and c_l is the chance level. The sum in eq. (6.3) was truncated at $\tau = 15$, as the performance was negligible for higher delays $\tau > 15$ for the network size $N = 150$.

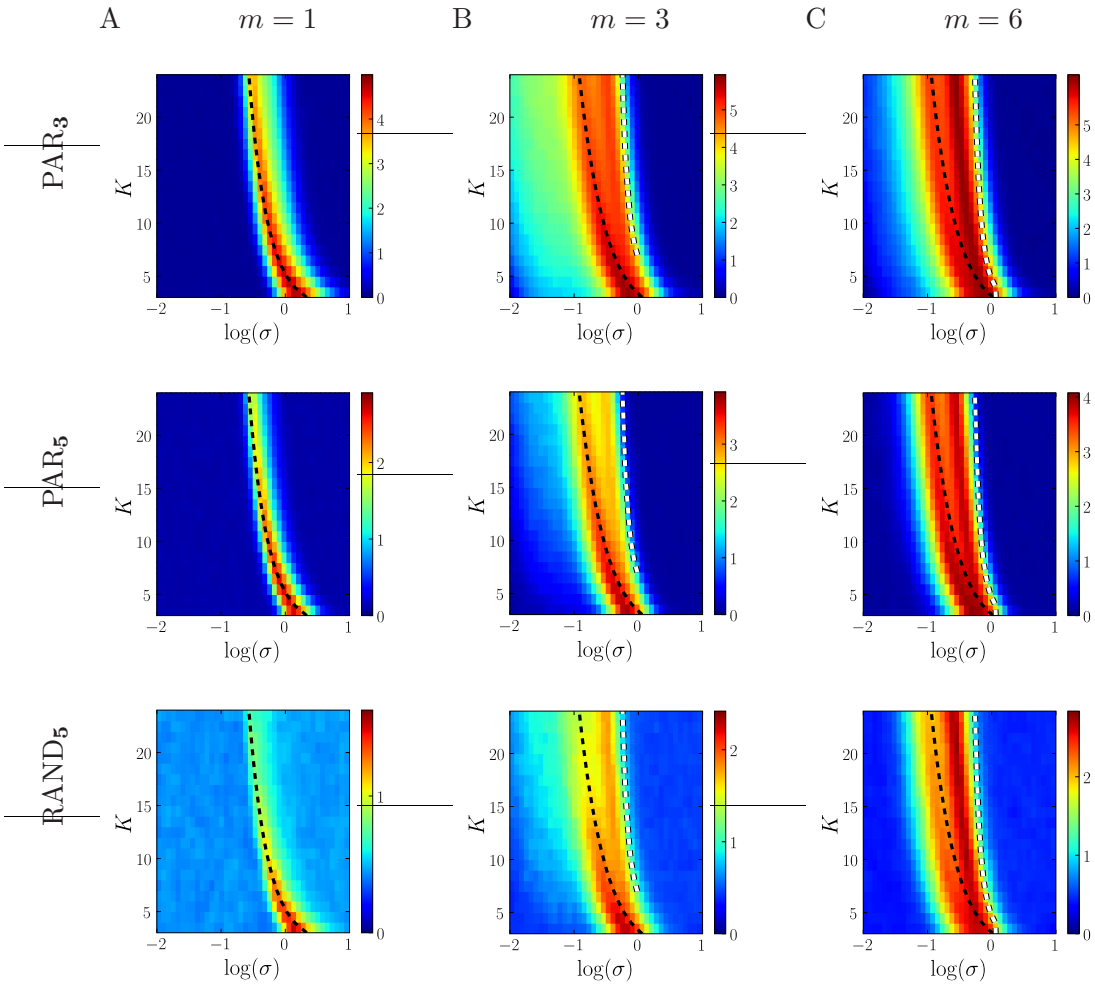


Figure 6.4: The performance $p_{\text{exp}}(C, \text{PAR}_3)$ (top row), $p_{\text{exp}}(C, \text{PAR}_5)$ (middle row), and $p_{\text{exp}}(C, \text{RAND}_5)$ (bottom row) for three different state resolutions $m = 1$ (left column A), $m = 3$ (center column B) and $m = 6$ (right column C) is plotted as a function of the network in-degree K and the weight STD σ. Further the zero crossing of the largest Lyapunov exponent λ (the critical line, black dashed line) as well as of the second largest one λ_2 (for $m \neq 1$, white dashed line) are shown. The networks size is $N = 150$, the results $p_{\text{exp}}(C, \text{PAR}_5)$ have been averaged over 20 circuits C, initial conditions and randomly drawn input time series of length 10^4 time steps. For $p_{\text{exp}}(C, \text{RAND}_5)$ results have been averaged over 50 random task of 5 bit, circuits C, initial conditions and randomly drawn input time series of length 10^4 time steps.

operates optimally for delays $\tau = 0, \ldots, 2$ on a given TASK_n and at chance level for other delays will have $p_{\mathrm{exp}}(C, \mathrm{TASK}_n) = 3$. Extensive numerical experiments indicate that the performance results for PAR_n can be considered quite representative for the general computational capabilities of a circuit C of the considered type as qualitatively very similar results were obtained for numerous classification tasks with two classes as well as for $p_{\mathrm{exp}}(C, \mathrm{RAND}_n)$ denoting the performance averaged over 50 randomly chosen functions f_T of n bits.

In Fig. 6.4 the performances $p_{\mathrm{exp}}(C, \mathrm{PAR}_3)$, $p_{\mathrm{exp}}(C, \mathrm{PAR}_5)$ averaged over 20 circuits C and $p_{\mathrm{exp}}(C, \mathrm{RAND}_5)$ averaged over 50 circuits C and random tasks for three different state resolutions $m = 1, 3, 6$ are shown. The results are plotted as functions of the network in-degree K and the logarithm[3] of the weight STD σ. Qualitatively very similar results were obtained for network graphs with binomial or scale-free distributed in-degree with average K (results not shown). The critical line, i. e. the location of the order-chaos phase transition, is also shown (dashed black line), which was determined by the root of the largest Lyapunov exponent λ given by the branching process approach outlined the previous section. Further the root of the second largest Lyapunov exponent λ_2 is plotted (dashed white line). The critical line predicts the zone of optimal performance well for $m = 1$, but is less accurate for ESNs with $m = 3, 6$. The root of λ_2 gives a quite reliable "upper bound" for the weight STD σ, i. e. all networks with a σ for which $\lambda_2 > 0$ are too chaotic to exhibit good performance measures p_{exp}. One can see that for ESNs with low state resolutions ($m = 1, 3$), networks with a small in-degree K reach a significantly better peak performance than those with high in-degree. The effect disappears for a high state resolution ($m = 6$). This phenomenon is consistent with the observation that network connectivity structure is in general an important issue if the reservoir is composed of binary or spiking neurons but less important if analog neurons are employed. The performance landscapes for analog networks ($m = \infty$) exhibit qualitatively the same key feature as the ones of high resolution networks (high performance can be found for all in-degrees by scaling σ) but differ quantitatively from the latter (the region of high performance is generally broader, results not shown). Note that for $m = 3, 6$ we see a bifurcation in the zones of optimal performance which is not observed for the limiting cases of ESNs and LSMs.

6.4 Phase Transitions in Quantized ESNs

In this section we examine the phase transition between ordered and chaotic dynamics in quantized ESNs in order to explain the difference between binary and high resolution reservoirs shown in Fig. 6.4. It was often hypothesized that systems with high computational power in recurrent networks are located in a parameter regime near the critical line, i. e. near the phase transition between ordered and chaotic behavior (see, e. g. (Legenstein & Maass, 2007b) for a review; compare also the performance with the critical line in Fig. 6.4). Starting from this hypothesis, we investigated whether the network dynamics of binary networks near this transition differs qualitatively from the

[3] All logarithms are taken to the basis 10, i. e. $\log = \log_{10}$ if not stated otherwise.

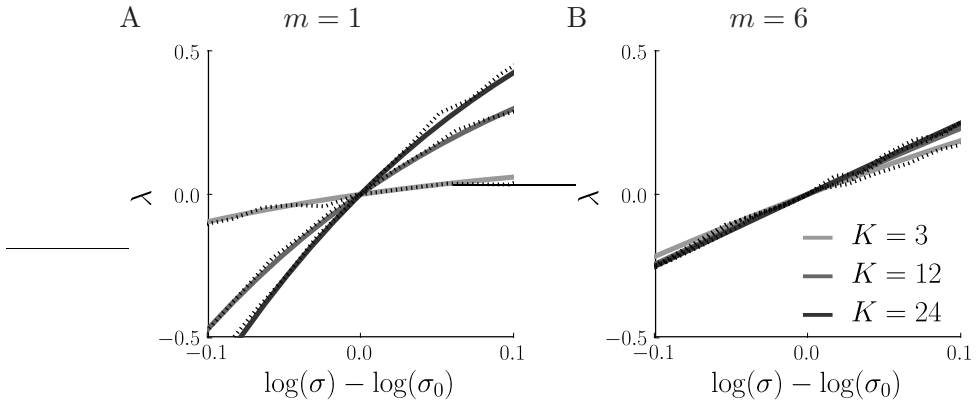


Figure 6.5: Phase transitions in binary networks ($m = 1$) differ from phase transition in high resolution networks ($m = 6$). The branching process approximation λ of the largest Lyapunov exponent is plotted as a function of the STD of weights σ for in-degrees $K = 3$ (light gray), $K = 12$ (Gray), and $K = 24$ (dark Gray). Further the corresponding finite-size estimations λ_{exp} (evaluated for $N = 150$) are shown (dotted black). In order to facilitate comparison, the plot for each K is centered around $\log(\sigma_0)$ where σ_0 is the STD of weights for which λ is zero (i. e. σ_0 is the estimated critical σ value for that K). The transition sharpens with increasing K for binary reservoirs (panel A), whereas it is virtually independent of K for high resolution reservoirs (panel B).

one of high resolution networks. We analyzed the network properties by considering the Lyapunov exponent λ approximated by the branching process approach introduced above. However, we did not only determine the critical line (i. e. the parameter values where the estimated Lyapunov exponent crosses zero), but also considered its values nearby. For a given in-degree K, λ can then be plotted as a function of the STD of weights σ (centered at the critical value σ_0 of the STD for that K). This was done for binary ($m = 1$, Fig. 6.5A) and high resolution networks ($m = 6$, Fig. 6.5B) with in-degrees $K = 3, 12$, and 24. Interestingly, the dependence of λ on the STD σ near the critical line is qualitatively quite different between the two types of networks. For binary networks the transition becomes much sharper with increasing in-degree K which is not the case for high resolution networks. These observations are confirmed by investigation of the numerically determined Lyapunov exponent λ_{exp} (plotted as dashed lines in Fig. 6.5) which agrees accurately with λ in the considered parameter regime.

How can this sharp transition between ordered and chaotic dynamics of binary ESNs with high in-degree K explain their reduced computational performance? The tasks considered in this article require some limited amount of memory which has to be provided by the reservoir. Hence, the network dynamics has to be located in a regime where memory about recent inputs is available and past input bits do not interfere with that memory. Intuitively, an effect of the sharper phase transition could be stated in the following way. For low σ (i. e. in the ordered regime), the memory needed for the task is not provided by the reservoir. With increasing σ, the memory capacity increases,

but older memories interfere with recent ones, making it more difficult for the readout to extract the relevant information. This intuition is confirmed by an analysis which was introduced in (Legenstein & Maass, 2007a) and which we applied to our setup. We estimated two measures of the reservoir, the so called "kernel-quality" and the "generalization rank", both being the rank of a matrix consisting of certain state vectors of the reservoir. These two measures quantify two complementary properties of a reservoir with respect to the target function to be learned by the readout. For both measures, one defines N different input streams $u_1(\cdot), \ldots, u_N(\cdot)$ and computes the rank of the $N \times N$ matrix, the state matrix, whose columns are the circuit states resulting from these input streams. The difference between the kernel-quality and the generalization rank arises from the choice of the input streams. For the kernel-quality, one chooses input streams which differ strongly with respect to the target function (e. g. streams that belong to different target classes). Since different input streams can only be separated by a readout if they are represented by the reservoir in a diverse manner, it is desirable that the rank of the state matrix is high in this case. For the generalization rank, one chooses similar input streams (again with respect to the target function). The rank of this state matrix should be small. The generalization rank can be related via the VC-dimension (Vapnik, 1998) to the ability of the reservoir-readout system to generalize from the training data to test data of the system (see (Legenstein & Maass, 2007a) for details). In general, a high kernel-quality and a low generalization rank (corresponding to a high ability of the network to generalize) are desirable. A network in the ordered regime will however have low values on both measures while a chaotic network will have high values on both measures. By the use of these two separate measures, one can gain some insight into the different factors that determine computational power of a reservoir system, as we will see below.

To evaluate the kernel-quality of the reservoir, we randomly drew $N = 150$ input streams $u_1(\cdot), \ldots, u_N(\cdot)$ and computed the rank of the $N \times N$ matrix whose columns were the circuit states resulting from these input streams.[4] Intuitively, this rank measures how well the reservoir represents different input streams. The generalization rank was evaluated as follows. We randomly drew N input streams $\tilde{u}_1(\cdot), \ldots, \tilde{u}_N(\cdot)$ such that the last three input bits in all these input streams were identical.[5] The generalization rank is then given by the rank of the $N \times N$ matrix whose columns are the circuit states resulting from these input streams. Intuitively, the generalization rank with this input distribution measures how strongly the reservoir state at time t is sensitive to inputs older than three time steps. The rank measures calculated in this way thus have predictive power for computations which require memory of the last three time steps.

Fig. 6.6A and D show the difference between the two measures as a function of $\log(\sigma)$ and K for binary networks and high resolution networks respectively. The plots show that the peak value of this difference is decreasing with K in binary networks, whereas it is independent of K in high resolution reservoirs, reproducing the observations in the

[4] The initial states of all neurons were iid. uniformly over $\mathcal{S}_m$. The rank of the matrix was estimated by singular value decomposition on the network states after 15 time steps of simulation.

[5] First, we drew each of the last three bits $\tilde{u}(13), \ldots, \tilde{u}(15)$ independently from a uniform distribution over $\{-1, 1\}$. For each input stream $\tilde{u}_i(1), \ldots, \tilde{u}_i(15)$ we drew $\tilde{u}_i(1), \ldots, \tilde{u}_i(12)$ independently from a uniform distribution over $\{-1, 1\}$ and set $\tilde{u}_i(t) = \tilde{u}(t)$ for $t = 13, \ldots, 15$.

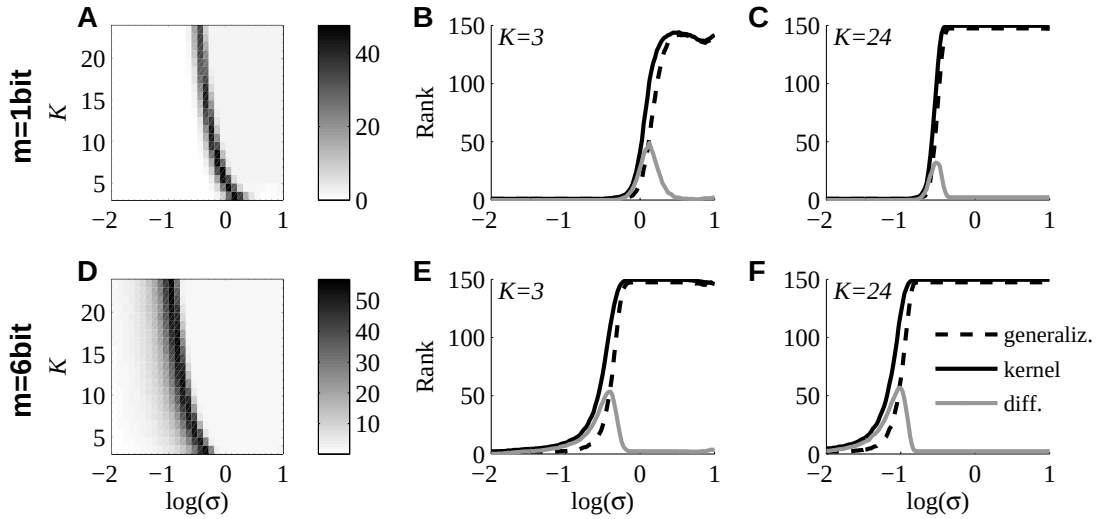


Figure 6.6: Kernel-quality and generalization rank of quantized ESNs of size $N = 150$. Upper plots are for binary reservoirs ($m = 1$), lower plots for high resolution reservoirs ($m = 6$). A: The difference between the kernel-quality and the generalization rank as a function of the log STD of weights and the in-degree K. B: The kernel-quality (red), the generalization rank (blue) and the difference between both (black) for $K = 3$ as a function of $\log(\sigma)$. C: Same as panel B, but for an in-degree of $K = 24$. In comparison to panel B, the transition of both measures is much steeper. D,E,F: Same as panels A, B, and C respectively, but for a high resolution reservoir. All plotted values are means over 100 independent runs with randomly drawn networks, initial states, and input streams.

plots for the computational performance Fig. 6.4. A closer look for the binary circuit at $K = 3$ and $K = 24$ is given in Fig. 6.6B and 6.6C. When comparing these plots, one sees that the transition of both measures is much steeper for $K = 24$ than for $K = 3$, in agreement with the observed sharper phase transition illustrated in Fig. 6.5, which leads to a smaller difference between the measures. We interpret this finding in the following way. For $K = 24$, the reservoir increases its separation power very fast as $\log(\sigma)$ increases. However the separation of past input differences increases likewise and thus early input differences cannot be distinguished from late ones. This reduces the computational power of binary ESNs with large K on the considered tasks. In comparison, the corresponding plots for high resolution reservoirs (Figs. 6.6E and 6.6F) show that the transition shifts to lower weight STDs σ for larger K, but apart from this fact the transitions are virtually identical for low and high K values. Comparing Fig. 6.6D with Fig. 6.4C, one sees that the rank measure does not accurately predict the whole region of good performance for high resolution reservoirs. It also does not predict the observed bifurcation in the zones of optimal performance, a phenomenon that is reproduced by the mean-field predictor introduced in the following section.

6.5 Mean-Field Predictor for Computational Performance

The question why and to what degree certain non-autonomous dynamical systems are useful devices for online computations has been addressed theoretically amongst others in (Bertschinger & Natschlaeger, 2004). There, the computational performance of networks of randomly connected threshold gates was linked to their separation property (for a formal definition see (Maass et al., 2002)): It was shown that only networks which exhibit sufficiently different network states for different instances of the input stream, i. e. networks that separate the input, can compute complex functions of the input stream. Furthermore, the authors introduced an accurate predictor of the computational capabilities for the considered type of binary networks based on the separation capability. The latter was numerically evaluated via a simple mean-field[6] approximation of the Hamming distance between different network states evolving from different input sequences.

Here we aim at constructing a mean-field predictor of computational performance for qESNs extending the approach of (Bertschinger & Natschlaeger, 2004) which was only viable for binary networks. We use the term "predictor" of computational power to indicate a quantity that strongly correlates with experimental measures of computational power (e. g. p_{exp}) for varying parameters K and σ at fixed m. A predictor in the above sense can be used to efficiently identify the dependence of the computational power on the network parameters and to gain theoretical insight into this dependence.

Instead of a straight forward generalization of the predictor presented in (Bertschinger & Natschlaeger, 2004) we make a somewhat different ansatz for two reasons. First we wish to simplify the form of the mean field predictor. Second, a straight forward generalization turned out to be computationally too expensive for quantized ESNs with a state resolution $m > 1$. Therefore we introduce a modified mean-field predictor which can be computed more efficiently and which still has all desirable properties of the one introduced in (Bertschinger & Natschlaeger, 2004).

Suppose the target output of the network at time t is a function $f_T \in F = \{f | f : \{-1, 1\}^n \to \{-1, 1\}\}$ of the n bits $u(t - \tau - 1), \ldots, u(t - \tau - n)$ of the input sequence u with delay τ as described in Section 6.3. In order to exhibit good performance on an arbitrary $f_T \in F$, pairs of inputs that differ in at least one of the n bits have to be mapped by the network to different states at time t. Only then will the linear classifier be able to assign the inputs to different classes (function values). In order to quantify this so-called separation property of a given network, we introduce the normalized distance $d(k)$: It measures the average distance between two networks states $\mathbf{x}^1(t) = (x_1^1(t), \ldots, x_N^1(t))$ and $\mathbf{x}^2(t) = (x_1^2(t), \ldots, x_N^2(t))$ arising from applying to the same network two input sequences u^1 and u^2 which only differ in the single bit at time $t - k$, i. e. $u^1(t - k) = -u^2(t - k)$ and $u^1(\tau) = u^2(\tau)$ for all $\tau \neq t - k$. Formally we

[6]The theoretical approach presented in (Bertschinger & Natschlaeger, 2004) is not a mean-field theory in the strict sense of Physics literature. However, due to the AA used in (Bertschinger & Natschlaeger, 2004), all network units receive recurrent inputs that are drawn (independently) from the *same* distribution. This is why we adopt the term "mean-field" and also apply it to our theoretical considerations.

define[7] the k-step input separation $d(k)$:

$$d(k) = \frac{1}{N} \left\langle \left\| \mathbf{x}^1(t) - \mathbf{x}^2(t) \right\|_1 \right\rangle_{C,u^1} . \tag{6.4}$$

The average $\langle . \rangle_{C,u^1}$ is taken over all inputs u^1 (the input u^2 is simply a function of u^1), all initial conditions of the network and all circuits C with given networks parameters N, m, σ, K. However, a good separation of the n relevant bits, i. e. $d(k) \gg 0$ for $\tau < k \leq n+\tau$, is a necessary but not a sufficient condition for the ability of the network to calculate the target function. Beyond this, it is desired that the network "forgets" all (for the target function) irrelevant bits $u(t-k)$, $k > n+\tau$ of the input sufficiently fast, i. e. $d(k) \approx 0$ for $k > n+\tau$. We use the limit $d(\infty) = \lim_{k\to\infty} d(k)$ to quantify this irrelevant separation which can be considered as noise wrt. the target function f_T. Hence, we propose the quantity p_∞ as a heuristic predictor for computational power:

$$p_\infty = \max \{ d(2) - d(\infty), 0 \} . \tag{6.5}$$

As the first contribution to p_∞ we chose $d(2)$ as it reflects the ability of the network to perform a combination of two mechanisms: In order to exhibit a high value for $d(2)$ the network has to separate the inputs at the time step $t-2$ and to sustain the resulting state distance via its recurrent dynamics in the next time step $t-1$. We therefore consider $d(2)$ to be a measure for input separation on short time-scales relevant for the target function.

The quantity p_∞ is calculated using a mean-field model similar to the one presented in (Bertschinger & Natschlaeger, 2004) which itself is rooted in the AA, the latter was already described briefly in Section 6.2. In the AA and with $N \to \infty$ all components of the difference $\mathbf{x}^1(t) - \mathbf{x}^2(t)$ appearing in (6.4) are iid.. Hence it is sufficient to determine the joint probability of a single network unit to be in state $s_i \in \mathcal{S}_m$ in the network receiving input u^1 and being in state $s_j \in \mathcal{S}_m$ in the network receiving input u^2. This probability is denoted as $q_{ij}(t, u^1, u^2)$. The diagonal elements $i = j$ of $q_{ij}(t, u^1, u^2)$ quantify the probability that the state of a unit is not affected by the difference in the inputs u^1 and u^2 whereas the off-diagonal elements $i \neq j$ quantify the probability that the state s_i of a unit is "flipped" to state s_j due to the different inputs u^1 and u^2. The separation $d(k)$ can be computed as:

$$d(k) = \sum_{i,j=0}^{2^m-1} q_{ij}(k, u^1, u^2) |s_j - s_j|. \tag{6.6}$$

The probabilities $q_{ij}(k, u^1, u^2)$ can be calculated iteratively using $q_{ij}(k-1, u^1, u^2)$ as distribution of the recurrent inputs at time step k (analogous to the method presented in (Bertschinger & Natschlaeger, 2004)). For high resolution networks however, there are 2^{2m} different elements $q_{ij}(t, u^1, u^2)$. In order to avoid this "combinatorial explosion" we introduce an approximation which we term separation approximation (SA). Consider a network unit which is in the state s_i. The state s_i can be uniquely represented by the

[7] For vectors $\mathbf{x} = (x_1, x_2, \ldots) \in \mathbb{R}^N$ we use the Manhattan norm $\|\mathbf{x}\|_1 := \sum_{i=1}^N |x_i|$

binary tuple $(B_0(s_i), \ldots, B_{m-1}(s_i)) \in \{0,1\}^m$ where $B_0(s_i)$ is the most and $B_{m-1}(s_i))$ the least significant bit of this binary representation. We can then define the probability $q^l_{\alpha,\beta}(k, u^1, u^2)$ for $\alpha, \beta \in \{0,1\}$ as the probability of the l^{th} bit B_l to be in state α in the network receiving input u^1 and $B_l = \beta$ in the network receiving input u^2. The SA consists in assuming the probability of $B_l(s_i)$ to be independent from the ones of $B_n(s_i)$ for $n \neq l$:

$$q_{ij}(k, u^1, u^2) \approx \prod_{l=0}^{m-1} q^l_{B_l(s_i),B_l(s_j)}(k, u^1, u^2).$$

This approximation neglects the statistical dependencies between the bits $B_l(s_i)$ and $B_n(s_i)$ for $n \neq l$ for the sake of computational efficiency. Trivially, the SA is exact for binary reservoirs. Numerical results suggest that for small and intermediate state resolutions $m \leq 5$, the SA is still quite accurate whereas for large $m \geq 5$ deviations from full simulations of the network are clearly visible. All further details of the calculation of p_∞ can be found in Appendix F.3.

It is worth noticing that in the AA as the weights w_{ij} are symmetric[8] the following relation holds:

$$q_{ij}(t, u^1, u^2) = q_{ij}(t, \hat{u}^1, \hat{u}^2)$$

where $\hat{u}^1$ and $\hat{u}^2$ are sequences with $\hat{u}^1(\tau)\hat{u}^2(\tau) = u^1(\tau)u^2(\tau)$ for all $\tau \in \mathbb{Z}$. Hence, flipping input bits in both input sequences u^1 and u^2 at *the same time steps* leaves the input separation $d(k)$ unaltered in the AA. Therefore in the AA, $d(k)$ can be determined with a single sample sequence u^1 without the need for averaging over different inputs as indicated in (6.6) where the average $\langle\cdot\rangle_{u^1}$ does not appear. This constitutes the main advantages of p_∞ over the the predictor defined in (Bertschinger & Natschlaeger, 2004), the so-called NM-separation. The NM-separation requires averaging the mean-field separation measure over input sequences resulting in a significantly larger computation time. Furthermore, the NM-separation is determined by three contributions whereas p_∞ only contains two terms $d(2)$ and $d(\infty)$ making the latter more simple and intuitive.

In Fig. 6.7 two examples of the evolution of $d(k)$ for state resolutions $m = 1, 3, 6$ with the parameters $\log(\sigma) = -0.45$ and $K = 3, 24$ are shown. With this choice of σ the network with in-degree $K = 3$ is in the ordered regime for all state resolutions $m \in \{1, 3, 6\}$ and the network with in-degree $K = 24$ is in the chaotic regime. The mean-field approximations of $d(k)$ are in quite good agreement for $m = 1, 3$ with the values for $d(k)$ determined by explicitly simulating the networks. For high resolution networks (here $m = 6$) visible errors occur due to the AA and the SA.

In Fig. 6.8 the predictor p_∞ is plotted as a function of the weight STD σ and the in-degree K for three different values of the state resolution $m \in \{1, 3, 6\}$. When comparing these results with the actual network performance p_{exp} plotted in Fig. 6.4 one can see that p_∞ serves as a reliable predictor for p_{exp} of a network for sufficiently small m. For larger values of m the predictor p_∞ starts to deviate from the true performance

[8] We call a random variable z symmetric if and only if $p(z) = p(-z)$.

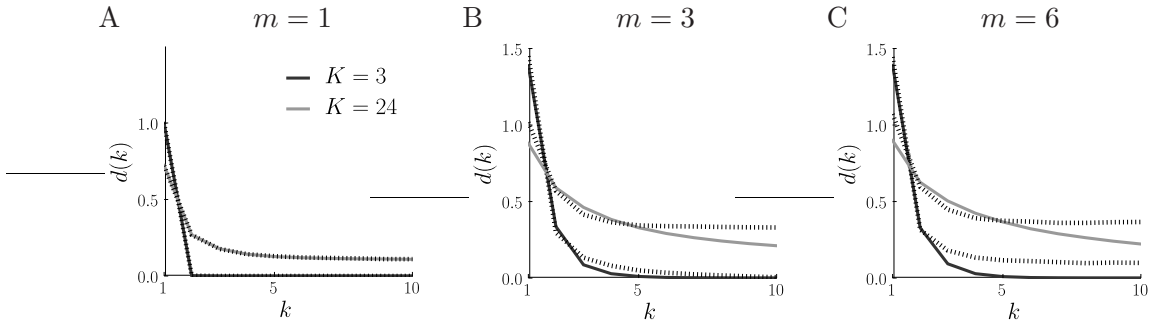


Figure 6.7: The k-step input separation measure $d(k)$ defined in (6.7) for networks with $N = 150$ units and $\log \sigma = -0.45$ with in-degree $K = 3$ (dark gray) corresponding to ordered dynamics and $K = 24$ (light gray) corresponding to chaotic dynamics determined by explicit simulations of the networks. The mean-field approximation of $d(k)$ is plotted as a dotted line showing good agreement for low state resolutions $m = 1, 3$ (panels A and B) and larger deviations for high state resolutions ($m = 6$ panel C).

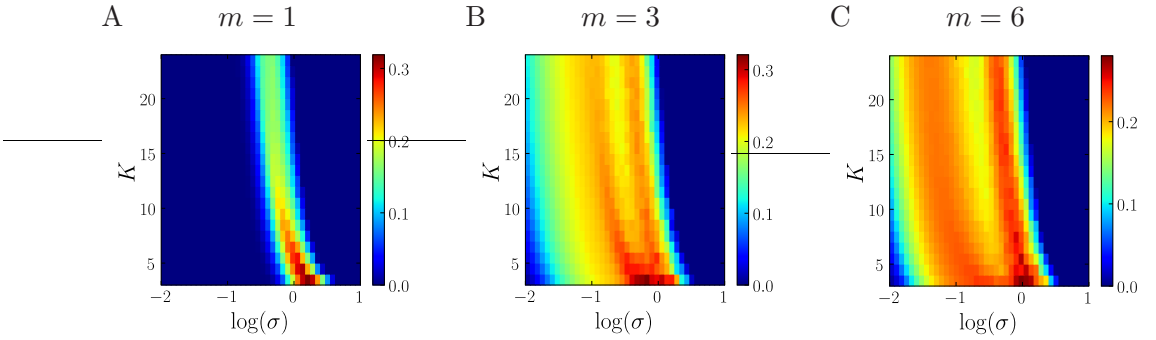


Figure 6.8: Mean-field predictor p_∞ for computational power for different state resolutions $m = 1$ (A), $m = 3$ (B), and $m = 6$ (C) as a function of the STD σ of the weights and in-degree K. Compare this result to the numerically determined performance p_{exp} plotted in Fig. 6.4.

while still capturing the interesting features of the performance landscape qualitatively. The dominant effect of the state resolution m on the performance discussed in Section 6.3 is well reproduced by p_∞: For $m = 1$ the in-degree K has a considerable impact, i. e. for large K maximum performance drops significantly. For high state resolutions however, for all values of K there exists a region in the parameter space exhibiting high performance.

The interplay between the two contributions $d(2)$ and $d(\infty)$ of p_∞ delivers insight into the dependence of the computational performance on the network parameters. A high value of $d(2)$ corresponds to a good separation of inputs on short time scales relevant for the target task, a property that is found predominantly in networks that are not strongly input driven, i. e. networks with relatively strong recurrent connection (large weight STD σ). A small value of $d(\infty)$ is a necessary condition for different inputs on which the target function assumes the same value to be mapped to nearby

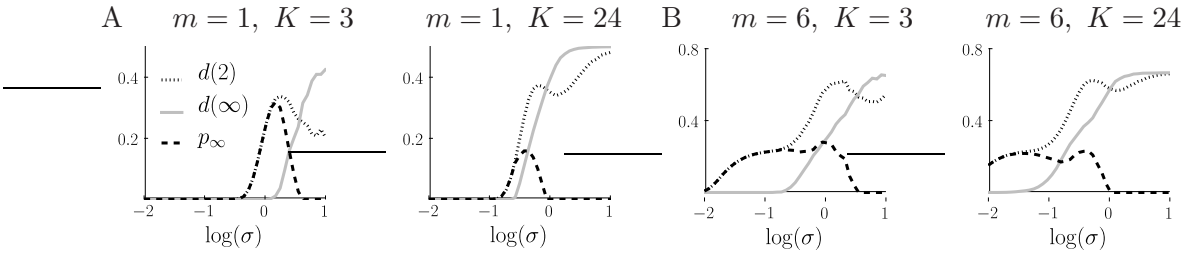


Figure 6.9: The contributions $d(2)$ (dotted line), $d(\infty)$ (light gray) of the mean-field predictor p_∞ and their difference (dashed black line) for different state resolutions $m = 1, 6$ as a function σ. The plots show slices of the 2D plots Fig. 6.8A and C for constant $K = 3, 24$. A: For $m = 1$ it can be seen that the region in $\log(\sigma)$-space with high $d(2)$ and low $d(\infty)$ is significantly larger for $K = 3$ than for $K = 24$. B: For $m = 6$ this region is roughly independent of K except a shift towards lower σ-values.

network states. Only then, a linear readout is able to assign them to the same class irrespectively of their irrelevant remote history. This condition is met for small weight STD σ. For $m = 1$, as can be seen in Fig. 6.9 the region in $\log(\sigma)$ space where both conditions for good performance are present, the region of intermediate σ, decreases for growing K. In contrast, for $m > 2$ a reverse effect is observed: for increasing K the parameter range for σ fulfilling the two opposing conditions for good performance grows moderately resulting in a large region of high p_∞ for high in-degree K. This observation is in close analogy to the behavior of the rank measure discussed in Section 6.4. Also note that p_∞ predicts the novel bifurcation effect also observed in Fig. 6.4.

6.6 An Annealed Approximation of the Memory Function

A quantity that has extensively been studied (White et al., 2004; Mayor & Gerstner, 2005) in order to characterize the ability of a given network to store information is the memory function defined in (Jaeger, 2002) (see also (Ganguli et al., 2008) for a novel alternative definition). In this section we show that the memory function for qESNs is tightly linked to the k-step separation $d(k)$ defined in equation (6.4). More precisely, the separation $d(k)$ can be used to formulate an upper bound on the memory function. For binary ($m = 1$) qESNs in the ordered regime, this upper bound turns out be very close to the true memory function while being computationally cheap to evaluate especially for networks with large system size N.

The memory function $m(k)$ defined in (Jaeger, 2002) which assumes values in $[0, 1]$ measures the ability of a network in conjunction with a linear readout to reconstruct the input signal $u(t - k)$ that was presented k time steps ago, where $m(k) = 1$ corresponds to a perfect reconstruction and $m(k) = 0$ to a readout output that is uncorrelated with the input $u(t - k)$. More precisely, the memory function is defined as:

$$m(k) := \frac{\mathrm{cov}(y(t), y_T(t))^2}{\mathrm{var}(y(t))\,\mathrm{var}(y_T(t))}. \tag{6.7}$$

Here $\mathrm{var}(\cdot)$ denotes the variance and $\mathrm{cov}(\cdot,\cdot)$ denotes the covariance of the arguments. The quantity $y(t) = (\sum_{i=1}^{N} \alpha_i x_i(t) + b)$ is the output of the linear readout at time t with weights $\alpha = (\alpha_1, \ldots, \alpha_N)$ and bias b and $y_T(t) = u(t-k)$ is the target output. The weights and the bias are learned by linear regression. According to the definition (6.7) the memory function measures the overlap between the readout output $y(t)$ and the target output $y_T(t)$. The memory function $m(k)$ is often numerically evaluated from the identity (see (Jaeger, 2002; White et al., 2004)):

$$m(k) \quad = \quad p_k^T A^{-1} p_k. \tag{6.8}$$

The matrix A with elements $A_{ij} = \mathrm{cov}(x_i(t), x_j(t))$ denotes the covariance matrix of the network state and $p_k = \mathrm{cov}(x(t), y_T(t))$ is the covariance vector between the network state and the target output. For networks with linear (thus analog) units many properties of the memory function can be characterized explicitly in terms of the connectivity matrix (see (Jaeger, 2002; White et al., 2004)). However, for networks of non-linear units little is known about the memory function, in general it has to be determined numerically by evaluating (6.8) which requires simulating the full network in order to estimate A and p_k.

For the special case of a binary input sequence u with $p(u(t) = +1) = p(u(t) = -1)$ as assumed in this chapter, the memory function can be bounded by using the k-step separation $d(k)$. The following relation is derived in Appendix F.4:

$$m(k) \quad \leq \quad \min\left\{ \frac{N^2}{4} \|A^{-1}\|_2 \; d(k)^2, 1 \right\}, \tag{6.9}$$

where $\|\cdot\|_2$ is the operator norm induced by the standard Euclidean norm. The upper bound presented in (6.9) is striking as it links the memory function $m(k)$ with the dynamical property of k-step input separation $d(k)$ allowing us to draw conclusions about $m(k)$ from the behavior of $d(k)$. As observed in numerical simulations $d(k)$ approximately decays exponentially[9] in the ordered regime implying that also $m(k)$ decays (at least) exponentially with a time constant that is half as large as the one for $d(k)$. This consideration also results in an upper bound for the scaling of the temporal capacity $k_C(N)$ with the network size N. In (White et al., 2004) $k_C(N)$ is defined as the smallest $k_0 \in \mathbb{N}$ such that for all $k > k_0$ the memory function of a network of size N is smaller than $1/2$, i. e. $m(k) < 1/2$. As can be easily seen from (6.9), $k_C(N) = \mathcal{O}(\log(N))$ given that $d(k)$ decays exponentially. Hence, the temporal capacity of all qESN networks in the ordered regime only grows logarithmically in contrast to e. g. linear networks with orthogonal connectivity matrices which exhibit an extensive growth $k_C(N) \propto N$ as show in (White et al., 2004). Another quantity of interest characterizing the capabilities of a circuit to store information is the memory capacity $\mathrm{MC}(N)$ which is defined as $\mathrm{MC}(N) := \sum_{k=1}^{\infty} m(k)$ (see (Jaeger, 2002)). In Fig. 6.10A $k_C(N)$ and $\mathrm{MC}(N)$ for $m = 1$ networks at the critical line are shown to exhibit a clearly logarithmic growth

[9] It is intuitive to conjecture that $d(k)$ should decay like $\exp(\lambda k)$ where λ is the Lyapunov exponent. However this is not trivial to show. Further investigation is required that addresses this interesting point.

with N over 1.5 decades. In Fig. 6.10B the performance measure p_{exp} is plotted for the three different task PAR_5, RAND_5 and SHIFT [10] as a function of N which also show logarithmic scaling with the system size N. In the chaotic parameter regime $d(k)$ decays towards a non-zero baseline and therefore the inequality (6.9) will not yield a useful upper bound on $m(k)$ as the latter always decays towards zero as numerically observed.

The inequality (6.9) can also be used to numerically estimate an upper bound for the memory function using the AA. This upper bound is computationally very efficient as its complexity is independent of the system size N and it turns out to be close to the true memory function. In the AA one can easily derive an expression for the term $\|A^{-1}\|_2$ as a function of the network connectivity parameters K and σ (see Appendix F.4.2):

$$\|A^{-1}\|_2 = 4\left(1 - \left(\Phi\left(\frac{2}{K^{1/2}\sigma}\right) - \Phi\left(-\frac{2}{K^{1/2}\sigma}\right)\right)^2\right)^{-1}. \tag{6.10}$$

Here $\Phi(x)$ denotes the cumulative probability distribution of a random variable distributed normally with unit variance. Combining (6.9) and (6.10) one can evaluate an upper bound for the memory function assuming that the AA is valid. However the accuracy of the mean-field approximation for $d(k)$ is of sufficient accuracy only for $m = 1$, hence the memory bound (6.9) is only of practical use for binary qESNs in the ordered regime. Three examples for $m(k)$ and for the upper bound (6.9) of binary qESNs with $N = 1000$ units with different parameter settings in the ordered regime are shown in Fig. 6.11. As can be seen, the distance between the memory function $m(k)$ and the upper bound (6.9) is varying with the weight STD σ and the in-degree K. It was numerically observed that the upper bound (6.9) was in general closer to $m(k)$ for networks whose parameters σ, K are "deeper" in the ordered dynamic regime. As mentioned above, in the chaotic regime the inequality (6.9) does not provide any sensible information on the memory function.

6.7 Sparse Network Activity and Computational Power

In the neocortex, the spiking (hence binary) neurons usually exhibit a high in-degree around 10^3 up to 10^4 (see (DeFelipe & Fariñas, 1992; Destexhe, Rudolph, & Pare, 2003)). Assuming the hypothesis that cortical circuits can be regarded (at least partially) as RC devices, the high in-degrees observed experimentally are in stark contrast to the findings described in the previous sections. As discussed above we would expect reservoirs consisting of binary units to be of low average in-degree as computational performance in the RC sense is best in this parameter regime. In the following section we show that this apparent inconsistence can be resolved by introducing sparse network

[10] The target output for the SHIFT_τ task is defined as $\text{SHIFT}_\tau(u,t) = u(t-\tau-1)$ and the performance is defined as $p_{\text{exp}}(C, \text{SHIFT}) = \sum_{\tau=0}^{\infty} \kappa(C, \text{SHIFT}_\tau)$. In contrast to the memory function a linear classifier is used to reconstruct the target signal.

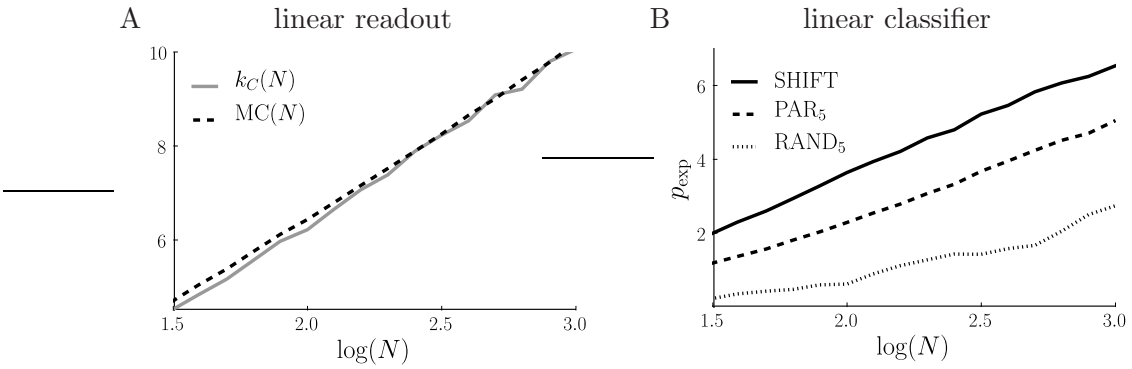


Figure 6.10: The computational capabilities of binary qESNs scale logarithmically with the network size N. The networks have the parameters $K = 3$ and $\log(\sigma) = 0.2$ and are thus close to the critical line. A: Scaling of the temporal capacity $k_C(N)$ (light gray) and the memory capacity $\mathrm{MC}(N)$ (dashed). According to the definition of the memory function a linear readout was used to reconstruct the target signal. B: Scaling of the experimental performance measures $p_{\text{exp}}(C, \mathrm{PAR}_5)$ (dashed) and $p_{\text{exp}}(C, \mathrm{RAND}_5)$ (dotted). Further the performance $p_{\text{exp}}(C, \mathrm{SHIFT})$ for the SHIFT task (solid line) is shown that consists in reconstructing the past input with a linear classifier. The experimental performance measures where determined as described in the caption of Fig. 6.4.

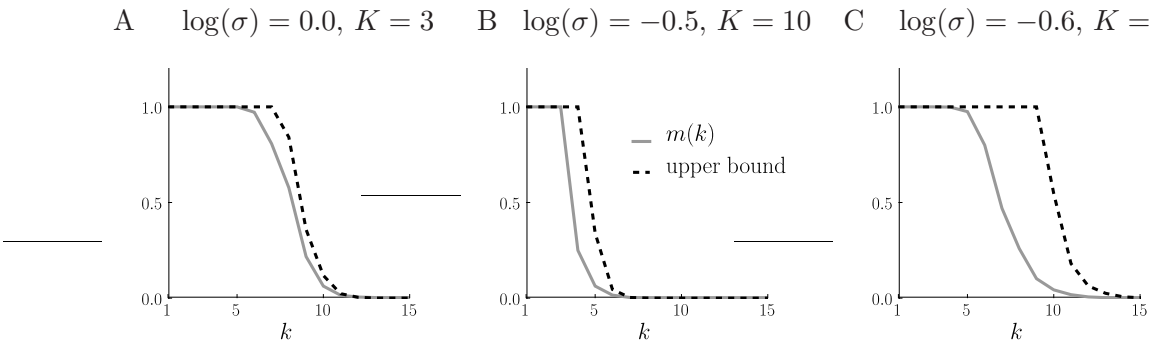


Figure 6.11: The memory function for three binary qESNs with $N = 1000$ units evaluated according to (6.8) is shown (light gray). Further, the upper bound given by equation (6.9) with the AA expression (6.10) for $\|A^{-1}\|_2$ is plotted (dashed). The qESNs were generated with the parameters $\log(\sigma) = 0.0$, $K = 3$ (panel A), $\log(\sigma) = -0.5$, $K = 10$ (panel B) and $\log(\sigma) = -0.6$, $K = 20$ (panel C). As can be seen, the distance between $m(k)$ and the upper bound varies depending on the network parameters. In general, the more "ordered" the network dynamics are, the "tighter" the upper bound (6.9) gets.

activity into the binary qESN model.

A characteristic property of the neural activity in the neocortex is that spikes are scarce events in time. Assuming that the refractory period of cortical neurons is in the millisecond range they could in principle emit spikes with a frequency of 100 Hz and more. However, the observed average firing rate of a cortical neuron is well below 10 Hz. It has often been suggested that this sparse activity is due to metabolic cost and energy constrains (see (W. B. Levy & Baxter, 1996; Laughlin, de Ruyter van Steveninck, & Anderson, 1998; Lennie, 2003)). The binary qESNs introduced above however are symmetric in the states $+1/2$ and $-1/2$ yielding equal probabilities to be in these two states. In order to mimic sparse network activity we augment the input $u(t)$ the state update equation (6.1) with a bias $b < 0$, i. e. we replace $u(t)$ by $u(t)+b$, which leads to a preferred "resting" state $-1/2$ and a less frequent "spiking" state $+1/2$. The probability for a unit to assume the value $1/2$ (to emit a "spike") can be evaluated in the AA to:

$$p_+ = p(x_i(t) = 1/2) = 1 - \frac{1}{2}\left(\Phi\left(\frac{2(-b-1)}{K^{1/2}\sigma}\right) + \Phi\left(\frac{2(-b+1)}{K^{1/2}\sigma}\right)\right). \tag{6.11}$$

In order to illustrate the impact of sparse activity on the computational performance we compare the measure p_{exp} of networks with different parameters σ and K at a given constant sparse activity p_+. To do so, the equation (6.11) is numerically solved for b yielding the required bias to achieve the given activity p_+ for a network with parameters σ and K. In Fig. 6.12 the resulting computational performance measure $p_{\text{exp}}(C, \text{PAR}_5)$ of binary qESNs for the five bit parity task PAR_5 is shown as a function of K and σ for three sparsity levels $p_+ = 0.3$ (panel A), $p_+ = 0.1$ (panel B) and $p_+ = 0.03$ (panel C). By comparison with Fig. 6.4 (where the activity is $p_+ = 0.5$) it can be observed that introducing a sufficiently sparse activity has a drastic effect on the computational performance landscape. In general the computational performance decreases with increasing sparsity of the network activity. The most striking effect however is that the parameter region of maximum performance is shifting towards higher connectivity values with increasing sparsity of the network activity. Hence, networks with sparser activity require a higher in-degree in order to exhibit good computational capabilities.

Given the above result (sparsely connected networks need higher activity to "work well") one might arrive at the intuitive hypothesis that for varying levels of connectivity and activity the average input to a single network unit is constant for reservoirs with high performance, i. e. that higher network activity can compensate for fewer input connections and vice versa. More precisely one might argue that for different mean activities p_+, the quantity $p_+ \cdot K_{\max}(p_+)$ is constant, where $K_{\max}(p_+) = \text{argmax}_K\, p_{\text{exp}}$ is the in-degree yielding the largest performance p_{exp} for the given activity p_+. However, results of numerical experiments we performed (data not shown) clearly indicate that this is not the case: The product $p_+ \cdot K_{\max}(p_+)$ varies strongly (by a factor of 1.6) for activities in the range $p_+ \in [0.1, 0.3]$. Networks with sparse activity need a higher in-degree than one would expect from the above hypothesis. Hence the optimal working point (in the RC sense) of a network cannot be determined by solely considering the average input to a single network unit. This insight is of relevance for the simulation of

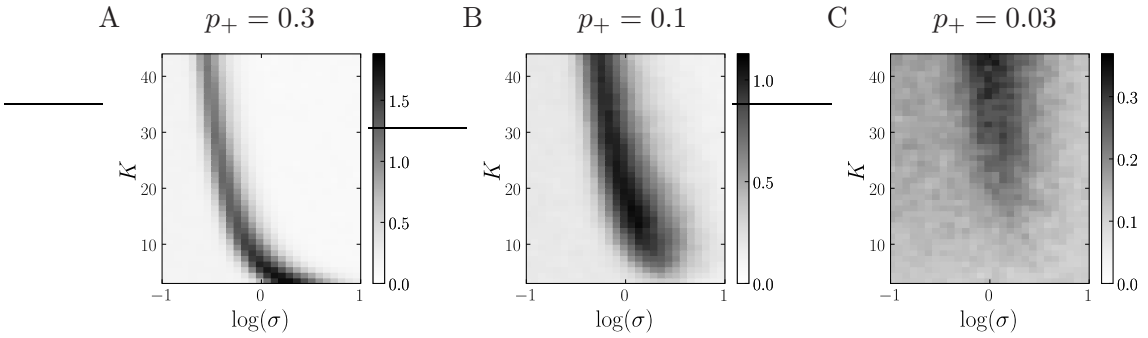


Figure 6.12: The peak computational performance of networks with sparser network activity is observed at increasing connectivity. Shown is the computational performance $p_{\text{exp}}(C, \text{PAR}_5)$ averaged over 20 random circuits C as a function of weight standard deviation σ and in-degree K for binary qESNs with sparse network activity $p_+ = 0.3$ (panel A), $p_+ = 0.1$ (panel B) and $p_+ = 0.03$ (panel C) caused by a bias.

cortical microcircuit models, where (due to limited computer resources) usually only low levels of connectivity are taken into account compensated by a higher network activity. Our analysis for the qESN model suggests that this approach might considerably change the computational properties of the simulated circuits, indicating that the working point of these circuits needs to be carefully tuned, possibly by adapting further parameters.

Unfortunately, the numerical results for qESNs with sparse network activity outline above cannot easily be reproduced by the mean-field predictor p_∞. Its numerical evaluation for networks with a non-vanishing bias is quite complex, as the bias destroys the symmetry of the update equation (6.1) which is explicitly taken advantage of for the calculation of p_∞. Without this symmetry the evaluation of p_∞ requires averaging over input sequences $u(\cdot)$ which renders the computation very time consuming.

Recapitulating, a possible explanation in the the RC framework for the high in-degrees experimentally found in cortical microcircuits is the sparse network activity which is a ubiquitous feature observed in cortex.

6.8 Discussion

In this chapter we introduced the qESN model that interpolates between the binary LSM and the continuous ESN. This non-autonomous network model exhibits ordered or chaotic dynamics, separated by a phase transition, depending on the weight and connectivity parameters. These dynamical regimes are reminiscent of the ones observed in binary (see e. g. (Derrida & Stauffer, 1986)) and in multi-state (see (Solé, Luque, & Kauffman, 2000)) Kauffman networks. In agreement with previous results ((Wolfram, 1984; Langton, 1990; Packard, 1988; Legenstein & Maass, 2007a; Bertschinger & Natschlaeger, 2004), but see (Mitchell, Hraber, & Crutchfield, 1993)) qENSs show optimal computational performance near the critical line, i. e. the order-chaos phase transition.

The qESN model for RC computations allowed the systematic investigation of a fundamental difference between LSMs and ESNs arising from the different nature of

its reservoir units: The difference in the influence of the network connectivity onto the computational capabilities. Our results clearly show that for binary and low resolution reservoirs the network connectivity, parametrized here by the in-degree K, has a profound impact on the phase transition and hence also on the maximal computational performance. For sparse connectivity (small K) a "gradual" phase transition is found characterized by a small gradient of the Lyapunov exponent around its root resulting in a high peek performance for a large region of the parameter space. For densely connected binary and low resolution networks the phase transition becomes "steep" indication a reduced parameter region which exhibits desirable critical dynamics. This effect results in a significantly reduced computational performance of densely connected binary and low resolution networks. The performance of high resolution networks however does not exhibit this drastic dependence on the in-degree K. These numerical observations can be understood by analyzing rank measures, assessing the kernel and generalization properties, as well as by analyzing an annealed approximation of the input separation of a given network. In the light of these two theoretical approaches the observed influence of connectivity on computational performance can be successfully explained in terms of separation of the input on short, task-relevant time scales versus separation on long, task-irrelevant time scales.

We emphasize that the experimental and numerical results indicating little influence of connectivity on computational performance for high resolutions qESNs are based on the assumption that there is no inherent spatial structure of the networks and of the input, a situation often occurring in typical machine learning applications of RC systems. In particular we assumed in this study that all units receive the one-dimensional input $u(t)$ and the readout also gets input from all network units. If however the network exhibits a spacial structure, e. g. the set of network units receiving input and the set of units projecting to the readout are disjoint and "distant", different connectivity schemes may well influence the performance of high resolution qESNs and analog ESNs.

It should be noted that the effect of quantization cannot be emulated by additive or multiplicative iid. or correlated Gaussian noise on the output of analog neurons. The noise just degrades performance homogeneously and the differences in the influence of the in-degree observed for varying state resolutions cannot be reproduced. Thus, this type of noise is qualitatively different from the so-called "quantization noise" introduced by discretizing the state space of the network units.

The results presented in this chapter which emphasize the importance of a gradual order-chaos phase transition offer a possible explanation why ESNs are more often used in engineering applications than LSMs. Although these two RC systems can have a very similar performance on a given task (Verstraeten et al., 2007), it is significantly harder to create a LSM consisting of spiking neurons operating in a desirable dynamic regime i. e. at the edge-of-chaos. In order to archive this, the excitation and inhibition in the network need to be finely balanced to avoid quiescent or epileptic activity. For ESNs this is not the case, there is usually a broad parameter range in which the ESN performs well. This difference reveals the need especially regarding LSMs for homeostatic control mechanisms and/or unsupervised learning rules that bring and keep dynamic reservoir in a close-to-optimal working regime replacing (possibly sub-optimal) ad-hoc param-

eter settings. This kind of unsupervised dynamic reservoir optimization has become quite standard for ESNs ((Triesch, 2007; Schrauwen, Wardermann, Verstraeten, Steil, & Stroobandt, 2008), and see (Lukosecicius & Jaeger, 2007) for a review), for LSMs interesting steps in this direction have been undertaken amongst others in (Natschlaeger, Bertschinger, & Legenstein, 2005; Lazar, Pippa, & Triesch, 2007; Joshi & Triesch, 2008).

We have shown that the k-step separation $d(k)$ can also be used to efficiently compute an upper bound for the memory function of binary networks with ordered dynamics. Previously only the memory function of linear networks was studied in detail (White et al., 2004; Jaeger, 2002; Ganguli et al., 2008) whereas theoretical results for nonlinear networks were missing. Given the numerical observation that $d(k)$ decays exponentially in the ordered regime, one can infer from the presented upper bound on the memory function that the information storage capabilities of qESNs scale like $\mathcal{O}(\log(N))$ with the system size N. It was also shown numerically that this scaling holds for the performance on the representative classification tasks ($p_{\exp}(C, \mathrm{PAR}_5)$ and $p_{\exp}(C, \mathrm{RAND}_5)$) as well. These findings might indicate a trade-off for RC systems between memory capacity and kernel-quality. Two extreme examples can illustrate this consideration. On the one hand, delay-line networks as well as another special class of linear networks (the so-called orthogonal networks) exhibit a very good memory performance (their memory capacities scale like $\mathcal{O}(N)$) (White et al., 2004) while failing on classification tasks like PAR_n, AND_n (as they are not linearly separable) and with high probability (for large n) also failing on RAND_n. Hence their kernel-quality can be considered poor. On the other hand, the non-linear qESNs exhibit a comparably "homogeneous" performance over all tasks that were studied in this article indicating a good kernel-quality but only show logarithmic memory scaling. Formalizing and investigating this apparent trade-off might reveal deep insights into the art of RC system design.

We informally equated in this article reservoir computing systems with binary units with LSMs. Most work about LSMs is concerned with the modeling of cortical circuits and the reservoir consequently often consist of biologically inspired spiking neuron models such as integrate-and-fire type units. We did not explicitly simulate such biologically more realistic reservoirs, however our results for reservoirs with binary units show characteristic properties also observed in biological modeling. For example, the performance of spiking reservoirs (commonly termed liquids) also strongly depends on the in-degree distribution of the network (Haeusler & Maass, 2007). This indicates that the binary nature of spikes is an important factor in the network dynamics of cortical circuits, a feature included in binary qESNs but not present in mean-field or rate-coded models of biological circuits.

The finding that binary reservoirs have high performance exclusively for low in-degrees stands in stark contrast to the fact that cortical neurons feature high in-degrees of over 10^4. This raises the interesting question which properties and mechanisms of cortical circuits not accounted for in the qESN model may cause this discrepancy. We have shown that sparse network activity as observed in cortical networks is a suitable candidate mechanism as it indeed shifts the region of optimal performance to higher in-degree values in binary networks. Interestingly, the sparse activity regime has also been proposed as a good computational regime for ESNs (Schrauwen et al., 2008) and

can be easily attained using the unsupervised dynamic reservoir optimization techniques mentioned above.

Although sparse activity is a prominent property of neocortical circuits, it is not the only possible explanation for the topological discrepancy between cortical circuits and the optimal circuits identified by our analysis. For example, the transmission of information between neurons via synapses is know to be error-prone as e. g. vesicle release into the synaptic cleft is a stochastic process with little reliability (Tsodyks & Markram, 1997). This stochastic aspect of biological connectivity might well result in a considerably smaller "true" or effective in-degree that is closer to the parameter regime found to be optimal for binary qESNs.

6.9 Acknowledgments

This chapter is based on the paper *Connectivity, dynamics, and memory in reservoir computing with binary and analog neurons*, which was written by Lars Büsing (LB), Benjamin Schrauwen (BS) and Robert Legenstein (RL). LB developed the newtork model, performed the simulations and developed the mean-field predictors. BS investigated the differences between binary and analog reservoirs and designed the simulations together with LB and RL. RL contributed the rank measure analysis. LB, BS and RL wrote the manuscript.

Appendix A

List of Publications

Journal and Conference Papers

[7] Buesing, L., & Maass, W. (2010). A Spiking Neuron as Information Bottleneck. *Neural Computation*, *22*(9), 1961–1992.

[6] Clopath, C., Buesing, L., Vasilaki, E., & Gerstner, W. (2010). Connectivity reflects coding: a model of voltage-based STDP with homeostasis. *Nature Neuroscience*, *13*(3), 344–352.

[5] Buesing, L., Schrauwen, B., & Legenstein, R. (2010). Connectivity, dynamics, and memory in reservoir computing with binary and analog neurons. *Neural Computation*, *22*(5), 1272-1311.

[4] Schrauwen, B., Buesing, L., & Legenstein, R. (2009). On computational power and the order-chaos phase transition in reservoir computing. In *Proc. of NIPS 2008, Advances in Neural Information Processing Systems* (Vol. 21, pp. 1425–1432). MIT Press. **Student Paper Award (Honorable Mentions)**

[3] Clopath, C., Ziegler, L., Vasilaki, E., Buesing, L., & Gerstner, W. (2008). Tag-trigger-consolidation: A model of early and late long-term-potentiation and depression. *PLoS Computational Biology*, *4*(12).

[2] Buesing, L., & Maass, W. (2008). Simplified rules and theoretical analysis for information bottleneck optimization and PCA with spiking neurons. In *Proc. of NIPS 2007, Advances in Neural Information Processing Systems* (Vol. 20, pp. 193–200). MIT Press.

[1] Muller, E., Buesing, L., Schemmel, J., & Meier, K. (2007). Spike-frequency adapting neural ensembles: Beyond mean adaptation and renewal theories. *Neural Computation*, *19*(11), 2958–3010.

Conference and Workshop Abstracts

[7] Buesing, L., Bill, J., Habenschuss, S., Nessler, B., & Maass, W. (n.d.). Emergence of Bayesian computation in generic motifs of cortical microcircuits In *40th Annual Conference of the Society for Neuroscience.* (submitted)

[6] Clopath, C., Buesing, L., Vasilaki, E., & Gerstner, W. (n.d.). Why is connectivity in barrel cortex different from that in visual cortex? - A plasticity model. In *Proc. of Computational and Systems Neuroscience 2010.*

[5] Schrauwen, B., & Buesing, L. (2009). A Hierarchy of Recurrent Networks for Speech Recognition. In *Deep Learning for Speech Recognition and Related Applications (NIPS Workshop).*

[4] Clopath, C., Ziegler, L., Buesing, L., Vasilaki, E., & Gerstner, W. (n.d.). Tag-trigger-consolidation: A model of early and late long-term potentation and depression. In *Proc. of Computational and Systems Neuroscience 2009.*

[3] Clopath, C., Ziegler, L., Buesing, L., Vasilaki, E., & Gerstner, W. (2009). Modeling plasticity across different time scales: The TagTriC model. In *BMC Neuroscience* (Vol. 10, p. P192).

[2] Buesing, L., & Maass, W. (2008). Information Bottleneck Optimization with Spiking Neurons with Application to Predictive Coding. In *Principled Theoretical Frameworks for the Perception-Action Cycle (NIPS Workshop).*

[1] Clopath, C., Buesing, L., Vasilaki, E., & Gerstner, W. (2008). A unified voltage-based model for STDP, LTP and LTD. In *Proc. of Computational and Systems Neuroscience 2008.*

Comments and Contributions to Journal and Conference Papers

The paper *Spike-frequency adapting neural ensembles: Beyond mean adaptation and renewal theories* is a joint paper together with Eilif Muller, Johannes Schemmel and Karlheinz Meier. It was written during my Master's thesis at Heidelberg University and published in *Neural Computation*. It is not included in this thesis.

The paper *Tag-Trigger-Consolidation: A Model of Early and Late Long-Term-Potentiation and Depression* is a joint paper together with Claudia Clopath, Lorric Ziegler, Eleni Vasilaki and Wulfram Gerstner. It includes results obtained during my stay at the Ecole Polytechnique Fédérale de Lausanne in 2007 and it was published in *PLoS Computational Biology*. The paper is not included in this thesis.

The paper *Connectivity reflects coding: A model of voltage-based STDP with homeostasis* was written by Claudia Clopath (CC), myself (LB), Eleni Valsilaki (EV) and Wulfram Gerstner (WG). The plasticity model was developed by CC and LB. CC fitted the model to experimental data, designed and carried out the simulations. LB developed the link to the BCM rule and did the calculations for the expected weight change presented in Appendix B. EV participated in discussions. WG supervised the project and wrote most of the manuscript. This paper was published in *Nature Neuroscience* and is the basis for Chapter 2 of this thesis.

The paper *Simplified rules and theoretical analysis for information bottleneck optimization and PCA with spiking neurons* was written by LB and Wolfgang Maass (WM). The Information Bottleneck learning rule and its analysis was developed by LB. The simulations were designed by LB and WM and conducted by LB. LB and WM wrote the manuscript. This paper was published in the proceedings of *Advances in Neural Information Processing Systems (NIPS) 2007* and is the basis for Chapter 3 of this thesis. The approach proposed in this paper was extended in the paper *A Spiking Neuron as Information Bottleneck*, which was written by LB and WM. The model was developed by LB. The simulations were designed by LB and WM and conducted by LB. LB wrote the manuscript. This paper was published in *Neural Computation* and is the basis for Chapter 4 of this thesis.

The paper *On computational power and the order-chaos phase transition in reservoir computing*, was written by Benjamin Schrauwen (BS), LB and Robert Legenstein (RL). BS investigated the differences between binary and analog reservoirs and designed the simulations. LB developed the network model, performed the simulations and developed the mean-field predictor. RL contributed the rank measure analysis. LB, BS and RL wrote the manuscript. This paper was published in the proceedings of *Advances in Neural Information Processing Systems (NIPS) 2007*, where LB was awarded the *Student Paper Award (Honorable Mentions)* for it. It is the basis for Chapter 5 of this thesis. The ideas of the latter publication were extended in the paper *Connectivity, dynamics, and memory in reservoir computing with binary and analog neurons*, which was written by LB, BS and RL. LB developed the network model, performed the simulations and developed the mean-field predictors. BS investigated the differences between binary and analog reservoirs and designed the simulations together with RL and LB. RL contributed the rank measure analysis. LB, BS and RL wrote the manuscript. This paper was published in *Neural Computation* and is the basis for Chapter 6 of this thesis.

Appendix B

Connectivity reflects Coding: A Model of Voltage-Based Spike Timing-Dependent Plasticity

B.1 Neuron Model

In contrast to standard models of STDP, the plasticity model presented in this chapter involves the postsynaptic membrane potential $u(t)$. Hence, predicting the weight change in a given experimental paradigm requires a neuron model that describes the temporal evolution of $u(t)$. For this purpose we chose the adaptive Exponential Integrate-and-Fire (AdEx) model (Brette & Gerstner, 2005) with an additional current describing the depolarizing spike after potential (Badel et al., 2008). The neuron model is described by a voltage equation:

$$C\frac{d}{dt}u = -g_L(u - E_L) + g_L\Delta_T \exp\left(\frac{u - V_T}{\Delta_T}\right) - w_{ad} + z + I,$$

where C is the membrane capacitance, g_L the leak conductance, E_L the resting potential and I the stimulating current. The exponential term describes the activation of a rapid sodium current. The parameter Δ_T is called the slope factor and V_T the threshold potential (Brette & Gerstner, 2005). A hyperpolarizing adaptation current is described by the variable w_{ad} with dynamics:

$$\tau_{w_{ad}}\frac{d}{dt}w_{ad} = a(u - E_L) - w_{ad},$$

where $\tau_{w_{ad}}$ is the time constant of the adaption of the neuron and a the subthreshold adaptation. Upon firing the variable u is reset to a fixed value V_{reset} whereas w_{ad} is increased by an amount b. The main difference to the Izhikevich model (Izhikevich & Edelman, 2008) is that the voltage is exponential rather than quadratic allowing a better fit to data (Badel et al., 2008). The spike afterpotentials of the cells used in typical STDP experiments (Sjöström et al., 2001) have a long depolarizing spike after potential. We therefore add an additional current z which is set to a value I_{sp} immediately after a spike occurs and decays otherwise with a time constant τ_z:

$$\tau_z\frac{d}{dt}z = -z.$$

Finally, we model the refractoriness shown in pyramidal cells (Badel et al., 2008) with the adaptive threshold V_T. V_T is set to $V_{T_{max}}$ after a spike and decays to $V_{T_{rest}}$ with a time constant τ_{V_T} (see (Badel et al., 2008)), i.e.:

$$\tau_{V_T}\frac{d}{dt}V_T = -(V_T - V_{T_{rest}}).$$

Parameters for the neuron model are taken from (Brette & Gerstner, 2005) for the AdEx, τ_z is set to 40 ms in agreement with (Sjöström et al., 2001; Badel et al., 2008) and kept fixed throughout all simulations (see table B.1 A).

B.2 Plasticity Model

Since synaptic depression and potentiation take place through different pathways (O'Connor et al., 2005) our model features two separate additive contributions to the plasticity rule, one for LTD and another one for LTP.

For the LTD pathway, we assume that presynaptic spike arrival at synapse i induces depression of the synaptic weight w_i by an amount $-A_{\text{LTD}}\,[\overline{u}_-(t)-\theta_-]_+$ that is a function of the average postsynaptic depolarization $\overline{u}_-$. The brackets $[\,]_+$ indicate rectification, i.e. $[x]_+ = x$ for $x \geq 0$ and $[x]_+ = 0$ for $x < 0$. This implements experimental findings showing that postsynaptic depolarization must exceed a certain value θ^- to establish depression of a synapse (Artola et al., 1990) (see Fig. 2.1 H). The quantity $\overline{u}_-(t)$ is an exponential low-pass filtered version of the postsynaptic membrane potential $u(t)$ with a time constant τ_-:

$$\tau_-\frac{d}{dt}\overline{u}_-(t) = -\overline{u}_-(t) + u(t).$$

The variable $\bar{u}_-$ is an abstract variable which could, for instance, reflect the level of calcium concentration (Shouval et al., 2002) or the release of endocannabinoids (Sjöström et al., 2003), though such an interpretation is not necessary for our rule. The presynaptic spike train is described as a series of short pulses $X_i(t) = \sum_n \delta(t - t_i^n)$ at times t_i^n where i is the index of the synapse and n an index that counts the spike. Depression is modeled as the following update rule, see also Fig. 2.1:

$$\frac{d}{dt}w_i^- = -A_{\text{LTD}}(\bar{\bar{u}})\,X_i(t)\,[\overline{u}_-(t) - \theta_-]_+ \qquad \text{if } w_i > w_{\min}, \tag{B.1}$$

where $A_{\text{LTD}}(\bar{\bar{u}})$ is an amplitude parameter that is under the control of homeostatic processes (Turrigiano & Nelson, 2004). For slice experiments the parameter has a fixed value extracted from experiment. For the network simulations in presented in Fig. 2.5 to Fig. 2.8, we make it depend on the mean depolarization $\bar{\bar{u}}$ of the postsynaptic neuron, averaged over a time scale of 1 s. Equation (B.1) is a simple method to implement homeostasis. Other methods such as weight rescaling would also be possible (Turrigiano & Nelson, 2004). The time scale of 1 s is not critical (100 s or more would be more realistic for homeostasis), but convenient for the numerical implementation.

For the LTP part, we assume that each presynaptic spike at the synapse w_i increases

the trace $\bar{x}_i(t)$ of some biophysical quantity, which decays exponentially with a time constant τ_x in the absence of presynaptic spikes, similar to previous work (Gerstner et al., 1996; Pfister & Gerstner, 2006). The temporal evolution of $\bar{x}_i(t)$ is described by:

$$\tau_x \frac{d}{dt}\bar{x}_i(t) = -\bar{x}_i(t) + X_i(t),$$

where X_i is the spike train defined above. The quantity $\bar{x}_i(t)$ could for example represent the amount of glutamate bound to postsynaptic receptors (Pfister & Gerstner, 2006) or the number of NMDA receptors in an activated state. The potentiation of w_i is modeled by the following expression, which is proportional to the trace $\bar{x}_i(t)$ (see also Fig. 2.1):

$$\frac{d}{dt}w_i^+ = +A_{\text{LTP}}\,\bar{x}_i(t)\,[u(t)-\theta_+]_+\,[\overline{u}_+(t)-\theta_-]_+ \qquad \text{if } w_i < w_{\max}. \tag{B.2}$$

Here, A_{LTP} is a free amplitude parameter fitted to the data and $\overline{u}_+(t)$ is another low-pass filtered version of $u(t)$ similar to $\overline{u}_-(t)$ but with a shorter time constant τ_+ around 10 ms. Thus positive weight changes can occur if the momentary voltage $u(t)$ surpasses a threshold θ_+ and, at the same time the average value $\overline{u}_+(t)$ is above θ_-.

The final rule used in the simulation is described by the equation

$$\frac{d}{dt}w_i = -A_{\text{LTD}}(\bar{\bar{u}})\,X_i(t)\,[\overline{u}_-(t)-\theta_-]_+ + A_{\text{LTP}}\,\bar{x}_i(t)\,[u(t)-\theta_+]_+\,[\overline{u}_+(t)-\theta_-]_+, \tag{B.3}$$

combined with hard bounds $w_{\min} \le w_i \le w_{max}$. For network simulation, $A_{\text{LTD}}(\bar{\bar{u}}) = A_{\text{LTD}}\,\frac{\bar{\bar{u}}^2}{u_{ref}^2}$ where u_{ref}^2 is a reference value.

B.3 Analysis of Plasticity Model

We establish a quantitative link between the novel plasticity model (B.3) and the BCM plasticity model (Cooper et al., 2004) under the assumption of a spiking linear Poisson (LP) neuron model. In a LP model, input spike trains $X_j(t) = \sum_j \delta(t-t_j^f)$ are low-pass filtered and weighted to give a subthreshold potential $u^s(t) = \sum_j \int_0^\infty \epsilon(s)X_j(t-s)ds$ where $\epsilon(s)$ is the time course of an EPSP and u^s is measured with respect to the resting potential θ_-. The LP neuron generates spikes stochastically with firing intensity ν^{post} that is proportional (with parameter $1/\alpha$) to the subthreshold membrane potential u^s, hence the probability of firing in a short time between t and $t+\Delta$ is $P_F(t;t+\Delta) = \nu^{\text{post}}(t)\Delta = u^s(t)\Delta/\alpha$. If the LP neuron spikes at time t_f^{post}, we add a short voltage pulse $\beta\delta(t-t_f^{\text{post}})$ to the membrane potential. The total membrane potential is therefore

$$u(t) = u^s(t) + \beta Y(t) + \theta_-, \tag{B.4}$$

where $Y(t) = \sum_f \delta(t-t_f^{\text{post}})$ denotes the spike train of the postsynaptic neuron and β is the integral weight of spikes. To illustrate the significance of β suppose that in a hypothetical experiment of 100 ms duration we find a single triangular action potential with amplitude $120[mv]$ and $1[ms]$ duration at half-maximum, and otherwise the voltage is constant at a value of $2[mv]$ above rest, then the *mean* voltage averaged over this $100[ms]$

period is $\int_0^{100[ms]} u(t)dt/100[ms] = 2\,\mathrm{mv} + 1.2\,\mathrm{mv} + \theta_-$ so that the weight parameter β in (B.4) should have a value of 1.2 mV. By construction the expected number of spikes of the LP neuron is equal to its instantaneous rate $< Y > (t) = \nu^{\mathrm{post}}(t) = u^s(t)/\alpha$. In the following derivation the time dependence of the variables is not explicitly denoted for the sake of simplicity (except for cases where this dependence is emphasized), e.g. $u(t)$ is abbreviated as u.

We assume that the neuron has N excitatory synapses which are stimulated by N presynaptic Poisson spike trains of rates $\nu^{\mathrm{pre}} = (\nu_1^{\mathrm{pre}}, \ldots, \nu_N^{\mathrm{pre}})$. Further, we assume that the presynaptic rates ν^{pre} are slowly varying quantities compared to the intrinsic time scales τ_+, τ_- of our plasticity model or those our neuron model (e.g. EPSP duration), which are all below 50 ms. This assumption explicitly results in the following simplifications: $\overline{\nu}^{\mathrm{pre}} \approx \nu^{\mathrm{pre}}$, $\overline{\nu}_+^{\mathrm{post}} \approx \nu^{\mathrm{post}}$ and $\overline{\nu}_-^{\mathrm{post}} \approx \nu^{\mathrm{post}}$. Here the following notation was used. For a variable q, $\overline{q}$ denotes a version of q low-pass filtered with the time constant τ_q, $\overline{q}_+$ and $\overline{q}_-$ correspond to the time constants τ_+ and τ_- respectively.

Using the LP model defined above in the plasticity rule (B.3) yields (if we suppress for the moment the dependence upon the homeostatic variable $\bar{\bar{u}}$)

$$\frac{d}{dt}w_i = -A_{\mathrm{LTD}}X_i(\overline{u}_-^s + \beta\overline{Y}_-) + A_{\mathrm{LTP}}\bar{x}_i\beta Y(\overline{u}_+^s + \beta\overline{Y}_+), \tag{B.5}$$

where it was used that all voltages are above resting potential θ_- since only excitatory inputs are considered and that only Y is above the firing threshold θ_+ since u^s is the subthreshold voltage. Now we take the average $\langle\cdot\rangle_{\mathrm{post}}$ over the postsynaptic spikes given the postsynaptic rate ν^{post}:

$$\left\langle \frac{d}{dt}w_i \right\rangle_{\mathrm{post}} = -(\alpha+\beta)A_{\mathrm{LTD}}X_i\overline{\nu}_-^{\mathrm{post}} + (\alpha+\beta)A_{\mathrm{LTP}}\bar{x}_i\beta\nu^{\mathrm{post}}\overline{\nu}_+^{\mathrm{post}}, \tag{B.6}$$

Here we have used $\left\langle Y(t)\overline{Y}_+(t)\right\rangle_{\mathrm{post}} = \nu^{\mathrm{post}}(t)\overline{\nu}_+^{\mathrm{post}}(t)$, which holds because $\overline{Y}_+(t)$ is not influenced by a possible spike at time t (just by spikes at times s with $s < t$) and it is thus uncorrelated with $Y(t)$ given ν^{post}. Applying the assumption of slowly varying input rates yields:

$$\left\langle \frac{d}{dt}w_i \right\rangle_{\mathrm{post}} = -(\alpha+\beta)A_{\mathrm{LTD}}X_i\nu^{\mathrm{post}} + (\alpha+\beta)A_{\mathrm{LTP}}\bar{x}_i\beta\nu^{\mathrm{post}}\nu^{\mathrm{post}}. \tag{B.7}$$

Taking the average $\langle\cdot\rangle_{\mathrm{pre}}$ over the presynaptic spikes given the presynaptic firing rates ν^{pre} neglecting spike-spike correlations (i.e. correlations between X_i and ν^{post} beyond rate correlations between ν^{pre} and ν^{post}) and applying the assumption of slowly varying input rates results in:

$$\begin{aligned}\left\langle \frac{d}{dt}w_i \right\rangle &= -(\alpha+\beta)A_{\mathrm{LTD}}\nu_i^{\mathrm{pre}}\nu^{\mathrm{post}} + (\alpha+\beta)A_{\mathrm{LTP}}\nu_i^{\mathrm{pre}}\beta\nu^{\mathrm{post}}\nu^{\mathrm{post}} \\ &= (\alpha+\beta)\beta A_{\mathrm{LTP}}\nu_i^{\mathrm{pre}}\nu^{\mathrm{post}}\left(\nu^{\mathrm{post}} - \frac{A_{\mathrm{LTD}}}{\beta A_{\mathrm{LTP}}}\right).\end{aligned} \tag{B.8}$$

Here $\langle\langle\cdot\rangle_{\text{post}}\rangle_{\text{pre}}$ was abbreviated as $\langle\cdot\rangle$. The factor $(\alpha+\beta)\beta$ can be interpreted as the learning rate in a rate-based plasticity model and $A_{\text{LTD}}/\beta A_{\text{LTP}} = \vartheta$ as the threshold for the transition of LTD to LTP in the 'quadratic' BCM model (Cooper et al., 2004). Since A_{LTD} depends on the slow time scale of homeostatic processes upon the long-term averaged potential $\bar{\bar{u}}$, the threshold ϑ is a sliding one. Just as in the BCM model (Cooper et al., 2004), our plasticity model responds to persistent periods of high activity with an increase in the threshold ϑ.

B.4 Calculations for the Functional Implications

We now apply the above results to the toy model of Fig. 2.4 and compare results of our rule to those of standard pair-based STDP. We assume that lateral connections are weak, so that neuronal firing is triggered by the injected current pulses. Since we focus on recurrent networks, the weights carry two indices (e. g. w_{ij} for a weight from neuron j to i) and the output spike train of neuron i is denoted as X_i.

B.4.1 Rate Coding

Since in Fig. 2.4 A, current pulses are injected with Poisson distributed timing at fixed rate, neurons i and j exhibit Poisson firing with rates ν_i and $\nu_j > 0$, respectively. From (B.8) it follows that all postsynaptic neurons with rate $\nu_i > \vartheta$ have increasing synaptic weights, while all those with $\nu_i < \vartheta$ have decreasing connections. After some time, connections reach their upper or lower bound so that all neurons with $\nu_i > \vartheta$ have input connections with all weights $w_{ij} = w_{\max}$, independent of j and all other neurons have no incoming connections, in agreement with the results of Fig. 2.4 A. Note that in this simulation the threshold ϑ was fixed since the homeostatic control of A_{LTD} was turned off.

Standard STDP. The weight change induced by STDP in the rate coding scenario (where pre- and postsynaptic activities are independent) is antisymmetric, i.e. if weight w_{ij} is potentiated w_{ji} is depressed. If the the STDP function is antisymmetric with respect to time reversal, i.e. if the amplitude of potentiation at pre-post timing with time difference $|\Delta|$ is the same as the that of depression with reversed timing, then

$$\Delta w_{ij} = \int_{\text{experiment}} \left(\frac{d}{dt} w_{ij}\right) dt = -\Delta w_{ji}.$$

Hence, STDP cannot produce strong bidirectional weights if weights are initialized with medium strength. Note however that bidirectional connections would be possible if the amount of potentiation outweighs the amount of depression, that is, if the integral over the STDP window is positive.

B.4.2 Temporal Coding

We assume that N neurons are arranged on a circle and fire one after the other in a fixed order $n, n+1, ...$ with an interval of τ. We calculate the weight change Δw_{ij} per

firing cycle given that the system has undergone infinitely many firing cycles before. It is also assumed that the voltage trace consists only of the spike events, however, similar results can be obtained if the subthreshold current injection (that makes the neurons spike in the scenario of Fig. 4B) is taken into account.

Because of the cyclic activation pattern, we can evaluate the term $\overline{u}_{i-}$ at a presynaptic spike time of X_j:

$$\int_{\text{cycle}} \overline{u}_{i-} X_j \, dt \quad = \quad \beta \sum_{n=0}^{\infty} \exp\left(-\frac{(j-i) \bmod N \cdot \tau + nN\tau}{\tau_-}\right). \tag{B.9}$$

Hence the amount of depression per cycle is, after summation:

$$\text{Depression} \quad = \quad \beta A_{\text{LTD}} \cdot \exp\left(-\frac{(j-i) \bmod N \cdot \tau}{\tau_-}\right) \frac{1}{1-\exp\left(-\frac{N\tau}{\tau_-}\right)}. \tag{B.10}$$

For potentiation we have to evaluate $\overline{u}_{i+}\overline{x}_j$ at the time of a postsynaptic spike X_i. Under the above assumptions, the factors $\overline{u}_{i+}$ and $\overline{x}_j$ can be summed up separately, so that the amount of potentiation per cycle is:

$$\text{Potentiation} \quad = \quad A_{\text{LTP}}\beta^2 \frac{1}{\exp\left(\frac{N\tau}{\tau_+}\right)-1} \cdot \exp\left(-\frac{(i-j) \bmod N \cdot \tau}{\tau_x}\right) \frac{1}{1-\exp\left(-\frac{N\tau}{\tau_x}\right)}.$$

We note that the depression term scales like $\exp(-(j-i) \bmod N)$, which is largest if the postsynaptic neuron i spikes directly before presynaptic neuron j in the cycle. For the the potentiation term which scales like $\exp(-(i-j) \bmod N)$ it is just the other way round: It takes the largest value if the postsynaptic neuron i is directly after the presynaptic neuron j. Hence weights from n to $n+k$ (for small $k > 0$), i.e. those weights that are "in cycle direction" are potentiated, and weights "against cycle direction" are depressed. This results in a highly asymmetric connection pattern.

Standard STDP. The calculations are exactly as in the temporal coding case of the novel plasticity rule, except for the factor $\overline{u}_{i+}$ (which does not dependent on i or j and hence is just a global scaling). In this setup the novel rule and STDP behave equivalently.

B.5 Parameters and Data Fitting

For the plasticity experiments in slices, we take $\bar{\bar{u}} = u_{ref}$ as fixed and fit the parameters A_{LTD}. The total number of parameters of the plasticity model is then seven. For all data sets, except the one taken from (Ngezahayo et al., 2000), the threshold θ^- is set to the resting potential and θ^+ to the firing threshold of the AdEx model, i.e. $\theta^- = -70.6\,\text{mV}$ and $\theta^+ = -45.3\,\text{mV}$. The remaining five parameters τ_x, τ_-, τ_+, A_{LTD} and A_{LTP} are fitted to each data set individually by the following procedure. We calculate the theoretically predicted weight change $\Delta w_i^{\text{th},j}$ by integrating (analytically or numerically) (B.3) for a given experimental protocol j, as a function of the free parameters. We

then estimate the free parameters by minimizing the mean-square error E between the theoretical calculations and the experimental data $\Delta w_i^{\text{exp},j}$:

$$E = \sum_j \left(\Delta w_i^{\text{th},j} - \Delta w_i^{\text{exp},j} \right)^2 .$$

For the data set in hippocampus (Ngezahayo et al., 2000), we also fit the two parameters θ^- and θ^+ since completely different preparations and cell type were used. Moreover for this data set, the time constant τ_x is taken from physiological measurements given in (Bi & Poo, 2001) and fixed to the values of 16 ms. The parameters for the various experiments are summarized in Table B.1 B.

B.6 Protocols and Mathematical Methods

Voltage clamp experiment. (Fig. 2.1 H) The postsynaptic membrane potential was set in the simulations to a constant value u_{clamp} chosen from $-80[mv]$ to 0 mV while presynaptic fibers were stimulated with either 25 (blue line) or 100 pulses (red line) at 50 Hz. Due to voltage clamping, the actual value of the voltage u itself and the low-pass filtered versions $\bar{u}$ are constant and equal to u_{clamp}. Hence, the synaptic plasticity rule becomes $\frac{d}{dt} w_i = -A_{\text{LTD}}\, X_i(t)\, [u_{\text{clamp}} - \theta_-]_+ + A_{\text{LTP}}\, \bar{x}_i(t)\, [(u_{\text{clamp}} - \theta_-)(u_{\text{clamp}} - \theta_+)]_+$.

Frequency dependence experiment. (Fig. 2.2 B) Presynaptic spikes in the simulation were paired with postsynaptic spikes that were either advanced by 10 ms or delayed by -10 ms with respect to the presynaptic spike. This pairing was repeated 5 times with different frequencies ranging from 0.1 to 50 Hz. These 5 pairings were repeated 15 times at 0.1 Hz. However, the 5 pairings at 0.1 Hz were repeated only 10 times to mimic the experimental protocol from (Sjöström et al., 2001).

Burst timing-dependent plasticity. (Fig. 2.3 A) The presynaptic spike is paired $\Delta t = 10$ ms before (or $\Delta t = -10$ ms after) one, two or three postsynaptic spikes. The frequency of the burst is 50 Hz. The neuron receives 60 pairings at a frequency of 0.1 Hz. (Fig. 2.3 B) The presynaptic spike is paired with a burst of 3 action potentials ($\Delta t = 10$ ms and -10 ms), while the burst frequency varies from 20 to 100 Hz. (Fig. 2.3C) A presynaptic spike is paired with a burst of three postsynaptic action potentials with burst frequency of 50 Hz. The time Δt between the presynaptic spike and the first postsynaptic action potential varies from -80 to 40 ms. For a detailed description of the experiments see (Nevian & Sakmann, 2006).

Poisson input for functional scenarios.(Fig. 2.4-2.7) Poisson inputs are used in all the following experiments. They are generated by a stochastic process where the spike is elicited with a stochastic intensity ν.

Relation between connectivity and coding: Toy model.(Fig. 2.4) Weights of ten all-to-all connected neurons are initialized at 1, bounded between 0 and 3.

Weights evolve with the voltage-based rule (B.3) for 100 s. The model is compared to a canonical pair-based STDP model given by $\frac{d}{dt}w_i = -A^{\text{pair}}_{\text{LTD}}\, X_i\, \bar{y} + A^{\text{pair}}_{\text{LTP}}\, \bar{x}_i\, Y$, where Y is the postsynaptic spike train defined the same way as the presynaptic spike train X_i with a filter of the postsynaptic spikes $\bar{y}$ similar to $\bar{x}_i$. The parameters are chosen $A^{\text{pair}}_{\text{LTD}} = A^{\text{pair}}_{\text{LTP}} = 1e^{-5}$ for the amplitudes and τ_x for the time constant of $\bar{x}_i$ as well as for the time constant of the postsynaptic low-pass filter $\bar{y}$. Rate code: Neuron 1 fires at 2 Hz, neuron 2 at 4 Hz... neuron 10 at 20 Hz following Poisson statistics, i.e. short current pulses are injected to make the neuron fire with Poisson statistics at this frequency. Temporal code: Neurons fire successively every 20 ms, first neuron 1 fires then 20 ms later neuron 2 then... 10 then 1 etc, in a loop.

Rate coding in network simulation. (Fig. 2.5) Five hundred presynaptic Poisson neurons with firing rates ν_i^{pre} $(1 \leq i \leq 500)$ are connected to ten postsynaptic excitatory neurons. The input rates ν_i^{pre} follow a Gaussian profile, i. e. $\nu_i^{\text{pre}} = A \cdot \exp(-(i-\mu)^2/(2\sigma^2))$, with variance $\sigma = 10$ and amplitude $A = 30$ Hz. The center μ of the Gaussian shifts randomly every 100 ms between ten different positions equally distributed. Circular boundary conditions are assumed, i.e. neuron $i = 500$ is considered as neighbor of $i = 1$. Synaptic weights of the feedforward connections to the excitatory neurons are initialized randomly (uniformly in [0.5,2]) and hard bound are set to 0 and 3. The ten excitatory neurons are all to all recurrently connected with a starting synaptic weight of 0.25 (hard bounds set to 0 and 0.75). In addition, 3 inhibitory neurons are randomly driven by eight excitatory neurons and the feedforward inputs, they project on six excitatory neurons, also chosen randomly. Those random recurrent connections are fixed and have a weight equal to 1. The feedforward connections onto the inhibitory neurons are also fixed and chosen randomly between 0 and 0.5. The reference value is set to $u^2_{ref} = 60\,\text{mv}^2$ and the simulation time to 1000 s. Parameters are normally chosen as in Table B.1 B, visual cortex data, except for Fig. 2.5, where A_{LTP} and A_{LTD} where reduced by a factor 100.

Temporal coding in network simulation. (Fig. 2.7) Same setting as in the rate coding simulation except that the patterns are presented for 20 ms successively (from center position 50, to 100, to 150 etc in a circular manner). The reference value has been set to $u^2_{ref} = 80\,\text{mv}^2$. We use an asymmetry index calculated by the following procedure: first we relabel neurons according to the current position of their receptive field so that with the cyclic stimulation they get activated one after the other: $n \to n+1... \to n+k \to n-1 \to n$. We then compare the connections from n to $n+k$ to that from n to $n-k$ and compute $AS = \sum_k (w_{n,n+k} - w_{n,n-k})$, k = 1-3.

ICA-like computation - Orientation selectivity with natural images. (Fig. 2.8) Ten natural images have been taken from the benchmark given in (Olshausen & Field, 1996). A small patch of 16 by 16 pixels from any of the images is randomly chosen every 200 ms. Half of the time the image matrix is transposed, flipped around the vertical axis or the horizontal axis in order to remove any statistical orientation bias. After prewhitening, the inputs for the "ON" ("OFF") image are Poisson

spike trains generated by the positive (negative) part of the patch (with respect to a reference grey value reflecting the ensemble mean) with maximum frequency of 50 Hz. The 2x16x16 inputs are connected to one postsynaptic neuron. The initial weights are set randomly between 0 and 2 and hard bounds are set between 0 and 3. The connections follow the learning rule (B.3), where the reference value is set to $u_{ref}^2 = 50\,\mathrm{mV}^2$. Parameters are chosen as in Table B.1 B (visual cortex data) but A_{LTP} and A_{LTD} where reduced by a factor 10. Every 20 s an extra normalization is applied to equalize the norm of the "ON" weights to the one of the "OFF" weights (Miller, 1994).

Parameters	Value
C - membrane capacitance	281 pF
g_L - leak conductance	30 nS
E_L - resting potential	-70.6 mV
Δ_T - slope factor	2 mV
$V_{T_{rest}}$ - threshold potential at rest	-50.4 mV
$\tau_{w_{ad}}$ - adaptation time constant	144 ms
a - subthreshold adaptation	4 nS
b - spike triggered adaptation	0.805 pA
I_{sp} - spike current after a spike	400 pA
τ_z - spike current time constant	40 ms
τ_{V_T} - threshold potential time constant	50 ms
$V_{T_{max}}$ - threshold potential after a spike	-30.4 mV

Table B.1: Parameters for the neuron model.

Exper.	θ_- (mV)	θ_+ (mV)	A_{LTD} $(\mathrm{mV})^{-1}$	A_{LTP} $(\mathrm{mV})^{-2}$	τ_x (ms)	
VC*	-70.6	-45.3	$14e^{-5}$	$8e^{-5}$	15	
SC	-70.6	-45.3	$21e^{-5}$	$30e^{-5}$	30	...
HP	-41	-38	$38e^{-5}$	$2e^{-5}$	16	

	τ_- (ms)	τ_+ (ms)
	10	7
...	6	5

Table B.2: Plasticity rule parameters for the various experiments. VC stands for Visual Cortex cells (for experimental details see (Sjöström et al., 2001), * standard set of parameters), SC for Somatosensory Cortex cells (see (Nevian & Sakmann, 2006)) and HP for Hippocampal cells (see (Ngezahayo et al., 2000)). Bold numbers indicate the free parameters fitted to experimental data. Other parameters are set in advance to values based on the literature.

Appendix C

Simplified Information Bottleneck Optimization with Spiking Neurons

In this appendix the learning rules for Information Bottleneck optimization presented in chapter 3 are tested extensively by computer simulations of benchmark Information Bottleneck tasks defined in (Klampfl et al., 2009). Furthermore, we give details of the derivation of theses learning rules. A detailed thorough with the derivation of the learning rule presented in (Klampfl et al., 2009) is also given.

C.1 Further Simulation Results

In this section, the four IB tasks considered in in sections 5.1 to 5.4 of (Klampfl et al., 2009), here referred to as IB task 1 to 4, are solved using the spike-based and the rate-based learning rules presented in equations (8) and (9) in chapter 3. The tasks are described only shortly here, as a detailed description can be found in (Klampfl et al., 2009). The IB tasks introduced in section 5.5 in (Klampfl et al., 2009), where the target signal and the input were uncorrelated, could not be solve with the learning rules presented in chapter 3 as expected, since these learning rules only takes second order joint statistics of input X and target signal Y_T into account.

In the following simulations, the nonlinear neuron with refractory mechanism considered in (Klampfl et al., 2009) is used instead of the linear Poisson neuron, in order to demonstrate the applicability of the learning rules given in chapter 3 for more complex models. The firing probability density $g(t)$ is given by [1]:

$$g(t) = r_0 \log\left(1 + \exp\left(\frac{u}{u_0}\right)\right) R(t) \tag{C.1}$$

$$R(t) = \frac{\left(t - \hat{t} - \tau_{abs}\right)^2}{\tau_{ref}^2 + \left(t - \hat{t} - \tau_{abs}\right)^2} \Theta\left(t - \hat{t} - \tau_{abs}\right), \tag{C.2}$$

where Θ is the Heaviside step-function, $\hat{t}$ denotes the time of the last spike, τ_{abs}, τ_{ref}, u_0 and r_0 are constants > 0. The term $R(t)$ implements a relative and absolute refractory mechanism.

The numerical results are shown in Fig. C.1 (spike-based rule) and Fig. C.2 (rate-based rule). In general all results given in (Klampfl et al., 2009) in sections 5.1 to 5.4

[1] Mind the slightly different notation used in equations (2) to (4) in (Klampfl et al., 2009).

could be qualitatively reproduced, except for a minor difference in task 2 (described below). The general setup for the four IB tasks is a special case of the setup discussed in chapter 3.4: The 100 synapses are split into four equal-sized subgroups $G_1, \ldots, G_4$, all receiving Poisson spike trains. Spike trains for different subgroups are independent (except for task 3); spike trains in one subgroup all share the same statistics. The target signal is chosen to have mutual information with some of the subgroups.

Task 1 Extracting a single rate-modulation (task given in section 5.1 in (Klampfl et al., 2009)). The different subgroups receive Poisson spike trains with different rate modulations which are sinusoidal (G_1), piecewise constant (G_2), bursting (G_3) and constant (G_4). It was possible to extract the group G_i for $i = 1, 2, 3$ from the input and at the same time to depress the other groups, if the target signal was chosen to have the same rate modulation as G_i. Thus, the results given in 5.1 in (Klampfl et al., 2009) could be reproduced. In the example shown in Fig. C.1**A** and C.2**A**, Y_T had the same rate modulation as G_1.

Task 2 Extracting a time varying combination of rate modulations (task given in section 5.2 in (Klampfl et al., 2009)). All groups were driven by Poisson spike trains with independent, identically distributed rate modulations $r_l(t)$, which are piecewise constant. The target signal was generated with a rate modulation $(r_1(t)+r_2(t))/2$, archiving to extract the subgroups G_1 and G_2. After a time interval of 1500 s, this linear combination was changed to $(r_1(t) + r_3(t))/2$, resulting in the depression of G_2 and the potentiation of G_3, while G_1 remained potentiated. After another 1500 s the target signal was completely turned off. However, the synaptic distribution did not persist long after switching of the target signal, in contrast to the results reported in (Klampfl et al., 2009). This is due to the different terms D_{KL} and L_3 in the objective functions optimize in (Klampfl et al., 2009) and chapter 3 respectively. The term D_{KL} aims at archiving an optimal value for the total weight different from zero; thus if all weights are low, it has a global increasing effect. In contrast, L_3 establishes a pure decaying term; in the absence of the target signal all weights decay to zero. However, a learning rule of the form (3.8) or (3.9) in chapter 3, where the term corresponding to L_3 was replaced by D_{KL}, would not perform significantly better. A long time after turning off the target signal, only those principal components of the input X with the lowest eigenvalues λ_i would not be depressed, thus the weights would specialize one the uncorrelated part of the input (this can be seen from the deterministic equation (3.10) in chapter 3.

This is an inherent problem of the general IB task. If $I(Y, Y_T) = 0$ because $Y_T = 0$, the solution minimizing $I(X, Y)$ is $w = 0$. A possible way to circumvent this problem could be to introduce a learning rate α that is dependent on the average firing rate of the target signal Y_T.

Task 3 Extracting spike-spike correlations (task given in section 5.3 in (Klampfl et al., 2009)). The subgroups receive spike-spike correlated Poisson trains of correlation coefficient 0.5 (G_1), 0.2 (G_2) and 0.5 (G_3); the spike trains for G_4 are uncorrelated. Spike trains for G_1 and G_2 were mutually correlated but uncorrelated to spike

trains of G_3. The target spike train has correlations with G_1 and G_2, resulting in a potentiation of these two subgroups and in the depression of the remaining subgroups G_3 and G_4. The results given in section 5.3 in (Klampfl et al., 2009) could thus be reproduced.

Task 4 Extracting information with two target signals (task given in section 5.4 in (Klampfl et al., 2009)). Information in the first two groups was encoded in rate modulations of the form of low-pass filtered white noise (G_1) and bursting (G_2). Groups G_3 and G_4 have spike-spike correlations of coefficient 0.5, but are independent of each other. The target signal is given by two Poisson spike trains, the first sharing the same rate modulation as G_1 (additionally corrupted by additive white noise), the second having spike-spike correlation with G_3. The subgroups G_1 and G_3 were potentiated, the remaining subgroups were depressed, reproducing the results given in section 5.4 in (Klampfl et al., 2009).

C.2 Derivation of the Information Bottleneck Learning Rules

In this section the derivation of the spike-based and the rate-based Information Bottleneck learning rule equations (3.8) and (3.9) presented in chapter 3 is outlined; this derivation is in close analogy to the ones presented in (Toyoizumi et al., 2005) and (Klampfl et al., 2007). Important differences between the approach taken here and the approach presented in (Klampfl et al., 2007) are outlined in section C.5. We will only focus on the derivation of the learning rule for the Information Bottleneck optimization, as the rule (3.15) in chapter 3 for Principal Component Analysis (PCA) can be obtained by simple sign changes.

C.2.1 The Spike-Based Learning Rule

For the sake of simplicity the learning rule is derived in discrete time with time steps of equal distance Δt. The final rule in continuous time can be obtained by taking the limit $\Delta t \to 0$ in the end. The dependence on the time for all time varying quantities is denoted as a superscript, for example the membrane potential $u(.)$ at time step $k \in \mathbb{N}$, corresponding to time $t^k = k\Delta t$, is denoted as $u(t^k) = u^k$. For convenience, a maximal time step $K \in \mathbb{N}$ is assumed, the resulting rule however will be independent of K. Spike trains up to time step $k \leq K$ are written as k-tuples of random variables, for example $Y^k = (y^1, \ldots, y^k)$, $y^l \in \{0, 1\}$, with $y^l = 1$ if the postsynaptic neuron spike at t^l and $y^l = 0$ otherwise. Using the same notation, the j'th input spike train X_j^k for all $j \in \{1, \ldots, N\}$ and the target spike train Y_T^k are written as $X_j^k = (x_j^1, \ldots, x_j^k)$ and $Y_T^k = (y_T^1, \ldots, y_T^k)$ respectively. The membrane potential at time step k is given by the weighted sum of the presynaptic activities ν_j^k:

$$u^k = \sum_{j=1}^{N} w_j \nu_j^k = \sum_{j=1}^{N} w_j \sum_{n=1}^{k} \epsilon(t^k - t^n) x_j^n.$$

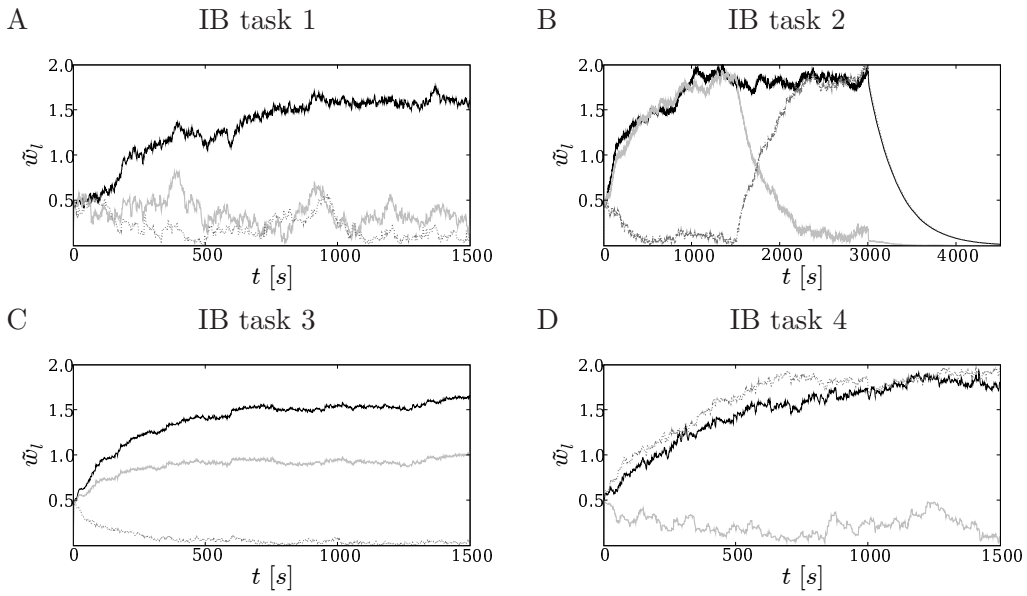


Figure C.1: The numerical result for the four IB tasks 1 to 4 described in the appendix C.1 corresponding to the figures A to D with the spike-based rule (3.8) in chapter 3. Shown is the temporal evolution of the average subgroup weight $\tilde{w}_l = 1/25 \sum_{j \in G_l} w_j$ for the first three subgroups G_1 (solid black line), G_2 (solid gray line) and G_3 (dotted dark gray line). For the sake of readability the trace for G_4 was not shown, as it always decays and is never potentiated. The results given in section 5.1 to 5.4 in (Klampfl et al., 2009) could be reproduced with the spike-based learning rule (3.8) in chapter 3.

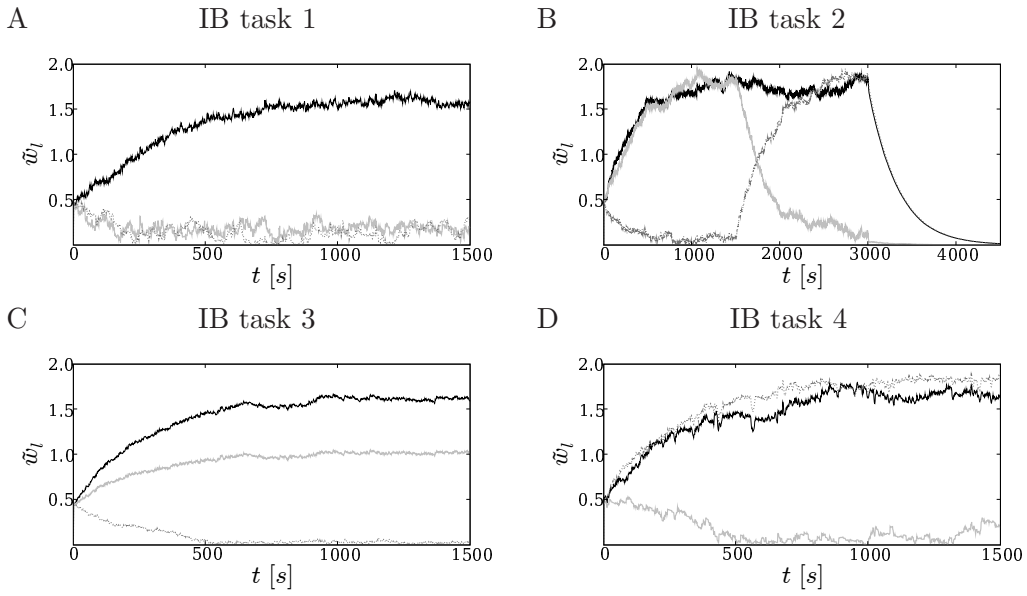


Figure C.2: Results for the IB tasks for the the rate-based rule (3.9). The figure is analogous to Fig. C.1

The quantity ρ^k is defined as the probability $P(y^k = 1|X^k)$ for the postsynaptic neuron to spike at time step k given the presynaptic input X^k up to time step k. It is proportional to the gain function g^k:

$$\rho^k = P(y^k = 1|X^k) = \Delta t g^k. \tag{C.3}$$

This definition only makes sense for a Δt which is chosen small enough, as $\rho^k \leq 1$ should be fulfilled. As the considered neuron model has no refractory mechanism, the output spikes are independent of each other, given the input X^K. Furthermore, the spike probability is independent of Y_T^K given X^K, thus:

$$P(y^k|X^k, Y_T^K, Y^{k-1}) = P(y^k|X^k, Y^{k-1}) = P(y^k|X^k). \tag{C.4}$$

The objective function, which is maximized for the IB optimization in chapter 3 is given by:

$$L = L_{\mathrm{IB}} + \beta_3 L_3 = \beta_1 I(Y^K, X^K) + \beta_2 I(Y^K, Y_T^K) + \beta_3 \sum_{i=1}^{N} w_i^2 = \sum_{h=1}^{3} \beta_h L_h. \tag{C.5}$$

The weight decay term $\beta_3 L_3$ in (C.5) is trivial to treat and as all operations necessary to obtain the final learning rule are linear, we concentrate only on the first two terms of (C.5); the contribution due to L_3 can simply be added to the result in the end. Thus, the objective function L is temporarily defined as:

$$L = L_{\mathrm{IB}} = \beta_1 I(Y^K, X^K) + \beta_2 I(Y^K, Y_T^K) = \sum_{h=1}^{2} \beta_h L_h. \tag{C.6}$$

Using the definition of the mutual information $I(Y^K, X^K)$ between the random variables Y^K and X^K, the first term L_1 can be written in the following way:

$$\begin{aligned} L_1 = I(Y^K, X^K) &= \sum_{X^K, Y^K} p(X^K, Y^K) \log\left(\frac{p(Y^K|X^K)}{p(Y^K)}\right) \\ &= \left\langle \log\left(\frac{P(Y^K|X^K)}{P(Y^K)}\right)\right\rangle_{Y_T^K, Y^K, X^K}. \end{aligned} \tag{C.7}$$

We use the following relation, known as the chain rule of probability theory (see (Cover & Thomas, 1991)):

$$P(Y^K|X^K) = \prod_{l=1}^{K} P(y^l|Y^{l-1}, X^l). \tag{C.8}$$

In analogy, one may write:

$$\begin{aligned} P(Y^K) &= \prod_{l=1}^{K} P(y^l|Y^{l-1}) \\ &= \prod_{l=1}^{K} \sum_{X^l} P(y^l|Y^{l-1}, X^l) P(X^l|Y^{l-1}) \\ &= \prod_{l=1}^{K} \Big\langle P(y^l|Y^{l-1}, X^l) \Big\rangle_{X^l|Y^{l-1}} . \end{aligned} \tag{C.9}$$

Plugging (C.8) and (C.9) into (C.7) yields:

$$\begin{aligned} L_1 = I(Y^K, X^K) &= \sum_{l=1}^{K} \left\langle \log \left(\frac{P(y^l|Y^{l-1}, X^l)}{\langle P(y^l|Y^{l-1}, X^l) \rangle_{X^l|Y^{l-1}}} \right) \right\rangle_{Y_T^K, Y^l, X^l} \\ &= \sum_{l=1}^{K} \Big\langle \mathcal{L}_1^l \Big\rangle_{Y_T^K, Y^l, X^l} \\ \mathcal{L}_1^l &:= \log \left(\frac{P(y^l|Y^{l-1}, X^l)}{\langle P(y^l|Y^{l-1}, X^l) \rangle_{X^l|Y^{l-1}}} \right) . \end{aligned}$$

The term L_2 can be written in a similar way:

$$L_2 = I(Y^K, Y_T^K) = \left\langle \log \left(\frac{P(Y^K|Y_T^K)}{P(Y^K)} \right) \right\rangle_{Y_T^K, Y^l, X^l} . \tag{C.10}$$

Using the chain rule of probability theory again:

$$P(Y^K|Y_T^K) = \prod_{l=1}^{K} P(y^l|Y^{l-1}, Y_T^K).$$

The equation (C.10) is then equal to:

$$\begin{aligned} L_2 = I(Y^K, Y_T^K) &= \sum_{l=1}^{K} \Big\langle \mathcal{L}_2^l \Big\rangle_{Y_T^K, Y^l, X^l} \\ \mathcal{L}_2^l &:= \log \left(\frac{P(y^l|Y^{l-1}, Y_T^K)}{\langle P(y^l|Y^{l-1}, X^l) \rangle_{X^l|Y^{l-1}}} \right) . \end{aligned}$$

Thus the objective function L given in (C.6) is of the form:

$$L = \sum_{l=1}^{K}\sum_{h=1}^{2}\beta_h \left\langle \mathcal{L}_h^l \right\rangle_{X^l,Y^l,Y_T^K} = \sum_{l=1}^{K} L^l. \tag{C.11}$$

$$L^l := \sum_{h=1}^{2}\beta_h \left\langle \mathcal{L}_h^l \right\rangle_{X^l,Y^l,Y_T^K} \tag{C.12}$$

As it can be seen in (C.11), the total objective function L can be written as a sum of per time step contributions L^l given by (C.12).

The online learning rule for the change of the synaptic efficacies w_j is obtained by a gradient ascent of the objective function L. In order to obtained a learning rule that is non-anticipating the weight change Δw_j^l of weight w_j^l at time-step l is chosen to be the gradient ascent of the per time step contribution L^l to the objective function:

$$\begin{aligned}\Delta w_j^l &= \alpha\left(\frac{\partial L^l}{\partial w_j}\right) \\ &= \alpha\sum_{h=1}^{2}\beta_h \partial_{w_j}\left\langle \mathcal{L}_h^l \right\rangle_{Y_T^K,Y^l,X^l}. \end{aligned}\tag{C.13}$$

The parameter α, called the learning rate, determines the step size of the gradient ascent. The gradient in w_j-direction can be calculated as:

$$\begin{aligned}\partial_{w_j}\left\langle \mathcal{L}_h^l \right\rangle_{Y_T^K,Y^l,X^l} &= \partial_{w_j}\sum_{Y_T^K,Y^l,X^l} P(Y_T^K,Y^l,X^l)\mathcal{L}_h^l \\ &= \partial_{w_j}\sum_{Y_T^K,Y^l,X^l} P(Y_T^K,X^l)P(Y^l|Y_T^K,X^l)\mathcal{L}_h^l &\text{(C.14)}\\ &= \sum_{Y_T^K,Y^l,X^l} P(Y_T^K,X^l)\partial_{w_j}\left(P(Y^l|Y_T^K,X^l)\mathcal{L}_h^l\right) &\text{(C.15)}\\ &= \left\langle \partial_{w_j}\mathcal{L}_h^l \right\rangle_{Y_T^K,Y^l,X^l} + \left\langle \mathcal{L}_h^l \partial_{w_j}\log P(Y^l|X^l)\right\rangle_{Y_T^K,Y^l,X^l}. &\text{(C.16)}\end{aligned}$$

The fact that the joint input distribution $P(Y_T^K,X^l)$ is independent form the weights of the postsynaptic neuron was used for going from (C.14) to (C.15).

It can be shown, that the terms $\langle\partial_{w_j}\mathcal{L}_h^l\rangle_{Y_T^K,Y^l,X^l}$ in (C.16) vanish, the proof for $\langle\partial_{w_j}\mathcal{L}_1^l\rangle_{Y_T^K,Y^l,X^l} = 0$ is given in equation (7) in the supporting text to (Toyoizumi et al., 2005). It remains to be shown that the contribution for $h = 2$ vanishes:

$$\left\langle \partial_{w_j}\mathcal{L}_2^l \right\rangle_{Y_T^K,Y^l,X^l} = \left\langle \partial_{w_j}\log P(y^l|Y^{l-1},Y_T^K)\right\rangle_{Y_T^K,Y^l} - \left\langle \partial_{w_j}\log P(y^l|Y^{l-1})\right\rangle_{Y^l} \tag{C.17}$$

We only outline the procedure for the first term of (C.17), as the second term can be

shown to be zero in a similar way:

$$\begin{aligned}\left\langle \partial_{w_j} \log P(y^l|Y^{l-1}, Y_T^K) \right\rangle_{Y_T^K, Y^l} &= \left\langle \left\langle \partial_{w_j} \log P(y^l|Y^{l-1}, Y_T^K) \right\rangle_{y^l|Y^{l-1}, Y_T^K} \right\rangle_{Y^{l-1}, Y_T^K} \\ &= \left\langle \sum_{y^l} P(y^l|Y^{l-1}, Y_T^K) \partial_{w_j} \log P(y^l|Y^{l-1}, Y_T^K) \right\rangle_{Y^{l-1}, Y_T^K} \\ &= \left\langle \partial_{w_j} \sum_{y^l} P(y^l|Y^{l-1}, Y_T^K) \right\rangle_{Y^{l-1}, Y_T^K} = 0.\end{aligned}$$

The last step follows from the normalization of probabilities.

Using these results, the gradient ascent (C.13) simplifies to:

$$\begin{aligned}\Delta w_j^l &= \alpha \sum_{h=1}^{2} \beta_h \left\langle \mathcal{L}_h^l \partial_{w_j} \log P(Y^l|X^l) \right\rangle_{Y_T^K, Y^l, X^l} \\ &= \alpha \sum_{h=1}^{2} \beta_h \left\langle \mathcal{L}_h^l C_j^l \right\rangle_{Y_T^K, Y^l, X^l}. \end{aligned} \tag{C.18}$$

The term C_j^l is defined as:

$$C_j^l =: \partial_{w_j} \log P(Y^l|X^l).$$

In order to further simplify the terms appearing in the preliminary learning equation (C.18), the probability of firing is written in the following way:

$$P(y^l|Y^{l-1}, X^l) = P(y^l|X^l) = \left(\rho^l\right)^{y^l} \left(1 - \rho^l\right)^{1-y^l}, \tag{C.19}$$

where ρ^l was defined in (C.3). Using (C.19), the term C_j^l may be cast into the form:

$$\begin{aligned} C_j^l &= \partial_{w_j} \log \left(\prod_{m=1}^{l} P(y^m|X^m) \right) \\ &= \sum_{m=1}^{l} \partial_{w_j} \left[y^m \log \rho^m - (1 - y^m) \log(1 - \rho^m) \right] \\ &= \sum_{m=1}^{l} \frac{(y^m - \rho^m)(\rho^m)'}{\rho^m (1 - \rho^m)} \nu_j^m, \end{aligned} \tag{C.20}$$

where $(\rho^m)'$ is the derivative of ρ^m with respect to u^m.

Now it is necessary to take a closer look at the $\mathcal{L}_h^l$. Using the relations, which follows

directly from (C.19):

$$\begin{aligned}\left\langle P(y^l|Y^{l-1},X^l)\right\rangle_{X^l|Y^{l-1}} &= \left\langle \rho^l\right\rangle^{y^l}_{X^l|Y^{l-1}}\left\langle 1-\rho^l\right\rangle^{1-y^l}_{X^l|Y^{l-1}}\\ P(y^l|Y^{l-1},Y_T^K) &= \left\langle P(y^l|X^l)\right\rangle_{X^l|Y^{l-1},Y_T^K}\\ &= \left\langle \rho^l\right\rangle^{y^l}_{X^l|Y^{l-1},Y_T^K}\left\langle 1-\rho^l\right\rangle^{1-y^l}_{X^l|Y^{l-1},Y_T^K}\end{aligned}$$

one may write the terms $\mathcal{L}_h^l$ as:

$$\begin{aligned}\mathcal{L}_1^l &= y^l\log\left(\frac{\rho^l}{\langle\rho^l\rangle_{X^l|Y^{l-1}}}\right)+(1-y^l)\log\left(\frac{1-\rho^l}{1-\langle\rho^l\rangle_{X^l|Y^{l-1}}}\right) \qquad \text{(C.21)}\\ \mathcal{L}_2^l &= y^l\log\left(\frac{\left\langle\rho^l\right\rangle_{X^l|Y^{l-1},Y_T^K}}{\langle\rho^l\rangle_{X^l|Y^{l-1}}}\right)+(1-y^l)\log\left(\frac{1-\left\langle\rho^l\right\rangle_{X^l|Y^{l-1},Y_T^K}}{1-\langle\rho^l\rangle_{X^l|Y^{l-1}}}\right).\end{aligned}$$

In general, the terms are of the form:

$$\mathcal{L}_h^l = y^l A_h^l + (1-y^l)B_h^l,$$

with the obvious defintions of A_h^l and B_h^l. In analogy to the approach taken in (Klampfl et al., 2007) and (Toyoizumi et al., 2005), the dependence of the terms A_h^l and B_h^l on Y^{l-1} via the expectation values $\left\langle\rho^l\right\rangle_{X^l|Y^{l-1}}$ and $\left\langle\rho^l\right\rangle_{X^l|Y^{l-1},Y_T^K}$ will be neglected[2], i. e. the following simplifications are used:

$$\begin{aligned}\left\langle\rho^l\right\rangle_{X^l|Y^{l-1}} &= \left\langle\rho^l\right\rangle_{X^l}\\ \left\langle\rho^l\right\rangle_{X^l|Y^{l-1},Y_T^K} &= \left\langle\rho^l\right\rangle_{X^l|Y_T^K}.\end{aligned}$$

This approximation is valid if only little information about the present firing probability ρ^l can be extracted from the recent firing history Y^{l-1}. This is the case for example if the temporal autocorrelation function of the input X^K decays faster than the average inter-spike interval of the output; this condition is fulfilled in most simulation examples discussed in chapter 3. Under this assumption $\mathcal{L}_1$, $\mathcal{L}_2$ are given by:

$$\begin{aligned}\mathcal{L}_1^l &= y^l\log\left(\frac{\rho^l}{\langle\rho^l\rangle_{X^l}}\right)+(1-y^l)\log\left(\frac{1-\rho^l}{\langle 1-\rho^l\rangle_{X^l}}\right)\\ \mathcal{L}_2^l &= y^l\log\left(\frac{\left\langle\rho^l\right\rangle_{X^l|Y_T^K}}{\langle\rho^l\rangle_{X^l}}\right)+(1-y^l)\log\left(\frac{1-\left\langle\rho^l\right\rangle_{X^l|Y_T^K}}{\langle 1-\rho^l\rangle_{X^l}}\right).\end{aligned}$$

Using this approximation, the following products, that appear in the learning rule (C.18),

[2]This simplification was done rather implicit in (Klampfl et al., 2007) and (Toyoizumi et al., 2005) by replacing the average $\langle.\rangle_{X^l|Y^{l-1}}$ by the low-pass filter $\overline{(.)}$, the latter being independent of the whole output spike train Y^K.

can be simplified:

$$\begin{aligned}
\left\langle C_j^l \mathcal{L}_h^l \right\rangle_{Y_T^K, Y^l, X^l} &= \left\langle \sum_{m=1}^{l} \frac{(\rho^m)'}{\rho^m(1-\rho^m)} \nu_j^m (y^m - \rho^m) \mathcal{L}_h^l \right\rangle_{Y^l, Y_T^K, X^l} \\
&= \left\langle \sum_{m=1}^{l} \frac{(\rho^m)'}{\rho^m(1-\rho^m)} \nu_j^m \left\langle (y^m - \rho^m) \mathcal{L}_h^l \right\rangle_{Y^l|Y_T^K, X^l} \right\rangle_{Y_T^K, X^l} \\
&= \left\langle \sum_{m=1}^{l} \frac{(\rho^m)'}{\rho^m(1-\rho^m)} \nu_j^m \left\langle (y^m - \rho^m)(y^l A_h^l + (1-y^l) B_h^l) \right\rangle_{Y^l|Y_T^K, X^l} \right\rangle_{Y_T^K, X^l} \\
&= \left\langle \sum_{m=1}^{l} \frac{(\rho^m)'}{\rho^m(1-\rho^m)} \nu_j^m \left\langle (y^m - \rho^m) y^l \right\rangle_{Y^l|Y_T^K, X^l} (A_h^l - B_h^l) \right\rangle_{Y_T^K, X^l} . \quad \text{(C.22)}
\end{aligned}$$

We now exploit the fact, the output Y^K is a pure Poisson process, for which the spikes are completely independent given the input X^l, i. e. $\langle y^l y^m \rangle_{Y^l|X^l} = \langle y^l \rangle_{Y^l|X^l} \langle y^m \rangle_{Y^l|X^l}$ and $\langle y^m \rangle_{Y^l|X^l} = \rho^m$ for $m \leq l$. This results in the following relation (where δ_{lm} is the Kronecker Delta[3]):

$$\langle y^l (y^m - \rho^m) \rangle_{Y^l|X^l, Y_T^K} = \left\langle \delta_{lm} \left(y^l - (\rho^l)^2 \right) \right\rangle_{Y^l|X^l, Y_T^K} . \quad \text{(C.23)}$$

By using (C.23), the equation (C.22), can be cast into the form:

$$\left\langle C_j^l \mathcal{L}_h^l \right\rangle_{Y_T^K, Y^l, X^l} = \left\langle \frac{(\rho^l)'}{\rho^l(1-\rho^l)} \nu_j^l \left(y^l - (\rho^l)^2 \right) (A_h^l - B_h^l) \right\rangle_{Y^l, Y_T^K, X^l} . \quad \text{(C.24)}$$

Plugging (C.24) into (C.18) yields the weight change:

$$\Delta w_j^l = \left\langle \alpha \left(y^l - (\rho^l)^2 \right) \frac{(g^l)'}{g^l(1-\rho^l)} \nu_j^l \left\{ \sum_{h=1}^{2} \beta_h (A_h^l - B_h^l) \right\} \right\rangle_{Y_T^K, Y^l, X^l} . \quad \text{(C.25)}$$

Here, $(g^l)'$ is the derivative of g^l wrt. the membrane potential. Using the linearity of g^l, this term is simply a constant $1/u_0$. The remaining expectation value over Y^l, Y_T^K, X_i^l in the learning rule (C.25) is assumed to be achieved by integrating over a long trial with small learning rate α, i. e. the rule is assumed to be self-averaging. Taking into account the contribution from the term L_3 in the objective function (C.5) one arrives at:

$$\Delta w_j^l = \alpha \frac{\left(y^l - (\rho^l)^2 \right)}{u^l(1-\rho^l)} \nu_j^l \left\{ \sum_{h=1}^{2} \beta_h (A_h^l - B_h^l) \right\} - \alpha \beta_3 w_j^l . \quad \text{(C.26)}$$

The continuous time limit $\Delta t \to 0$ can now be performed. As the terms B_h^l are of higher order in Δt compared to the A_h^l, they vanish; the same holds for the terms ρ^l and $(\rho^l)^2$,

[3]The Kronecker Delta δ_{lm} is zero for $m \neq l$ and one for $m = l$.

when compared to 1 and y^l respectively. Thus, the continuous time learning equation is given by:

$$\frac{d}{dt}w_j = \alpha\nu_j(t)\frac{Y(t)}{u(t)}\left\{\sum_{h=1}^{2}\beta_h A_h(t)\right\} - \alpha\beta_3 w_j(t). \tag{C.27}$$

Now the Lagrange multiplier β_1 is set to -1 (without loss of generality, as its absolute value can be absorbed into α), and β_2 and β_3 are identified with β and $-\lambda$ respectively, the latter appearing in the notation in equation (3.3) in chapter 3. Furthermore, the logarithm in the terms $A_h(t)$ can be expanded to linear order around unity in the case of small relative fluctuations of the membrane potential $u(t)$; this procedure is well justified for a large number of synapses, as fluctuations then tend to cancel out. The following approximations are thus applied:

$$A_1(t) = \log\left(\frac{u(t)}{\langle u(t)\rangle_{X^t}}\right) \approx \frac{u(t) - \langle u(t)\rangle_{X^t}}{\langle u(t)\rangle_{X^t}} \tag{C.28}$$

$$A_2(t) = \log\left(\frac{\langle u(t)\rangle_{X|Y_T}}{\langle u(t)\rangle_{X^t}}\right) \approx \frac{\langle u(t)\rangle_{X|Y_T} - \langle u(t)\rangle_{X^t}}{\langle u(t)\rangle_{X^t}}, \tag{C.29}$$

where $X^t := (X(\tau)|\tau \leq t)$. The terms $\langle u(t)\rangle_{X^t}$ and $\langle u(t)\rangle_{X|Y_T}$ are the continuous time limit of $\langle u^l\rangle_{X^l}$ and $\langle u^l\rangle_{X^l|Y_T^K}$ respectively.

Further following (Klampfl et al., 2007) and (Toyoizumi et al., 2005), the operator $\langle . \rangle_{X^t}$ appearing in (C.28) and (C.29) is replaced by the low-pass filter $\overline{(.)}$ with a large time constant τ_C (in simulations $\tau_C = 3\,\mathrm{s}$), i. e. the average of the membrane potential is estimated by:

$$\langle u(t)\rangle_{X^t} \longrightarrow \overline{u}(t) = \frac{1}{\tau_C}\int_{-\infty}^{t}\exp\left(-\frac{t-s}{\tau_C}\right)u(s)ds. \tag{C.30}$$

The final learning rule in continuous time, which will be referred to as the spike-based rule (as the output spike train $Y(t)$ explicitly appears), is given by:

$$\frac{d}{dt}w_j(t) = \alpha\frac{Y(t)}{u(t)}\nu_j(t)\left(-\frac{u(t)-\overline{u}(t)}{\overline{u}(t)} + \beta\frac{\langle u(t)\rangle_{X|Y_T} - \langle u(t)\rangle_{X^t}}{\overline{u}(t)}\right) - \alpha\lambda w_j(t). \tag{C.31}$$

It is realistic to assume that the information about the membrane potential $u(t)$, the low-pass filtered membrane potential $\overline{u}(t)$ and the presynaptic activity $\nu_j(t)$ is available at the site of the synapse j. Further, the time of a postsynaptic spike could be obtained the back-propagating action potential. In contrast, it is not straight-forward to see how the term $\langle u(t)\rangle_{X|Y_T}$ could possibly be sampled at the site of the synapse in an online process. Therefore this term is investigated in a more detailed way in section C.3.

C.2.2 The Rate-Based Learning Rule

In this section, the so-called rate-based learning rule accompanying the spike-based version (C.31) is derived. We start from equation (C.25):

$$\Delta w_j^l = \left\langle \alpha \left(y^l - (\rho^l)^2 \right) \frac{(g^l)'}{g^l(1-\rho^l)} \nu_j^l \left\{ \sum_{h=1}^{2} \beta_h (A_h^l - B_h^l) \right\} \right\rangle_{Y_T^K, Y^l, X^l} . \quad \text{(C.32)}$$

Instead of assuming the averaging operation $\langle . \rangle_{Y^l, X^l, Y_T^K}$ is carried out by the self-averaging process, it is split into two averages:

$$\langle . \rangle_{Y^l, X^l, Y_T^K} = \left\langle \langle . \rangle_{Y^l | X^l, Y_T^K} \right\rangle_{X^l, Y_T^K},$$

of which the inner one is performed immediately. Equation (C.32) then becomes:

$$\begin{aligned} \Delta w_j^l &= \left\langle \alpha \left\langle y^l - (\rho^l)^2 \right\rangle_{Y^l | X^l, Y_T^K} \frac{(g^l)'}{g^l(1-\rho^l)} \nu_j^l \left\{ \sum_{h=1}^{2} \beta_h (A_h^l - B_h^l) \right\} \right\rangle_{Y_T^K, X^l} \\ &= \left\langle \alpha \left(\rho^l - (\rho^l)^2 \right) \frac{(g^l)'}{g^l(1-\rho^l)} \nu_j^l \left\{ \sum_{h=1}^{2} \beta_h (A_h^l - B_h^l) \right\} \right\rangle_{Y_T^K, X^l} . \end{aligned}$$

Now, the remaining expectation value over X^l, Y_T^K is assumed to be achieved by the self-averaging with small learning rate α. Carrying out the same steps, that lead to the spike-based rule (C.31) yields the rate-based Information Bottleneck learning rule:

$$\frac{d}{dt} w_j(t) = \alpha \nu_j(t) \left(-\frac{u(t) - \overline{u}(t)}{\overline{u}(t)} + \beta \frac{\langle u(t) \rangle_{X|Y_T} - \langle u(t) \rangle_X}{\overline{u}(t)} \right) - \alpha \lambda w_j(t). \quad \text{(C.33)}$$

C.3 The Conditional Expectated Value of the Membrane Potential

As stated in section C.2.1, it is not obvious how the term $\langle u(t) \rangle_{X|Y_T}$, appearing in the spike-based learning rule (C.31) as well as in the rate-based version (C.33), could be estimated online at the site of the synapses. The term in question is obviously depending on the joint distribution of the stochastic processes $X(.)$ and $Y_T(.)$, which can be in general arbitrarily complex. To create some intuition for this problem, we shortly analyze the term $\langle u(t) \rangle_{X|Y_T}$ for a concrete example first before looking for a more general solution.

Consider the situation, where X_j for $j \in \{1, \ldots, N\}$ and Y_T are Poisson spike trains of constant rate μ_0 with spike-spike correlations of c_j, meaning that the two-point correlation function is given by:

$$\langle X_j(t) Y_T(s) \rangle = c_j \mu_0 \delta(t-s) + \mu_0^2. \quad \text{(C.34)}$$

Spike-spike correlated spike trains with the property (C.34) can be generated using the method outlined in (Gütig et al., 2003). Equation (C.34) implies that the probability density for X_j to spike at time t just depends on whether there was a spike in Y_T at time t or not. Thus:

$$\langle u(t)\rangle_{X|Y_T} = \sum_{j=1}^{N} w_j(t) \int_{\mathbb{R}} \epsilon(t-s) \langle X_j(s)\rangle_{X|Y_T(s)} \, ds. \tag{C.35}$$

As the distribution valued process Y_T only may assume two values 0 or $\delta(.)$ (i. e. a spike occurs or not), $\langle X_j(s)\rangle_{X|Y_T(s)}$ takes only two different values and (C.35) can be written in the following way:

$$\langle u(t)\rangle_{X|Y_T} = \sum_{j=1}^{N} w_j(t) \int_{\mathbb{R}} \epsilon(t-s) \left(a_j + b_j Y_T(s)\right) ds, \tag{C.36}$$

with coefficients a_j and b_j, which are functions of c_j. Thus, $\langle u(t)\rangle_{X|Y_T}$ for this example is a linear operator that acts on the target spike train Y_T. Higher order product terms of the form $Y_T(s_1)Y_T(s_2)\dots Y_T(s_m)$ only appear if the probability of a spike of X_j at time t depends on the joint occurrence of spikes of Y_T at times $s_1, \dots, s_n$.

In general, the quantity $\langle u(t)\rangle_{X|Y_T} / \overline{u}(t)$ is an operator on $Y_T(.)$, which will be denoted here as $F[.]$:

$$\frac{\langle u(t)\rangle_{X|Y_T}}{\overline{u}(t)} = F[Y_T](t). \tag{C.37}$$

In the following considerations it is assumed that F is independent of the weights as both the nominator and denominator of the fraction appearing in (C.37) are linear in the single weights w_j. Assuming the joint probability density of X and Y_T is time-invariant, the operator $F[.]$ also has this property. Furthermore it is a reasonable assumption that $F[.]$ has the fading memory property, as the the physical processes establishing the statistical dependence between X and Y_T surely take place on a finite time scale. Therefore, as shown in section C.4, it is possible to expand $F[Y_T]$ in a Volterra series of the form:

$$F[Y_T](t) = \sum_{n=0}^{\infty} \int_{\mathbb{R}} \cdots \int_{\mathbb{R}} \kappa_n(t-t_1, \dots, t-t_n) \prod_{i=1}^{n} Y_T(t_i) dt_i.$$

In chapter 3, we concentrate on the situation where the operator $F[.]$ can be well approximated by its linearization $F_1[.]$ plus a homogeneous term F_0:

$$F[Y_T](t) \approx F_0(t) + F_1[Y_T](t) = F_0(t) + \int_{\mathbb{R}} \kappa_1(t-t_1) Y_T(t_1) dt_1. \tag{C.38}$$

Using the linearized ansatz given by (C.38), the last term on the rhs of the learning rule

(C.31) can be written as:

$$\begin{aligned}\frac{\langle u(t)\rangle_{X|Y_T} - \langle u(t)\rangle_{X^t}}{\overline{u}(t)} &= \frac{\langle u(t)\rangle_{X|Y_T} - \left\langle \langle u(t)\rangle_{X|Y_T} \right\rangle_{Y_T}}{\overline{u}(t)} \\ &= F[Y_T](t) - \langle F[Y_T](t)\rangle_{Y_T} \\ &\approx F_1[Y_T](t) - \langle F_1[Y_T](t)\rangle_{Y_T}\,. \end{aligned} \tag{C.39}$$

Thus one arrives at:

$$\frac{\langle u(t)\rangle_{X|Y_T} - \langle u(t)\rangle_{X^t}}{\overline{u}(t)} = (\kappa_1 * Y_T)(t) - \langle (\kappa_1 * Y_T)(t)\rangle_{Y_T}\,,$$

where the asterisk $*$ denotes the convolution. In the chapter 3 it is assumed that the estimator defined by $\kappa_1 * Y_T$ is implemented by the concentration $u_T(t)$ of a neuromodulator, for positively correlated X and Y_T:

$$u_T(t) = (\kappa_1 * Y_T)(t) \tag{C.40}$$

$$\kappa_1(t) = \Theta(t)\exp\left(-\frac{t}{\tau_0}\right). \tag{C.41}$$

The biological mechanism behind equation (C.40) might be the following: Whenever a target signal spike arrives at time t_T^i, the concentration $u_T(t)$ at time t is increased by an amount $\kappa_1(t-t_T^i)$; the contributions of the spikes is assumed to add up linearly. In order to be biologically realistic, κ_1 has to fulfill the constrain, that it is not anticipating, i. e. $\kappa_1(\tau) = 0, \forall \tau < 0$, which is ensured by the Heaviside step function $\Theta(t)$. It is assumed that the value $u_T(t)$ is available at the site of the synapses at all times. The expectation value $\langle u_T(t)\rangle_{Y_T}$, in analogy the $\langle u(t)\rangle_{X^t}$, is replaced in the online rule by the low-pass filtered quantity $\overline{u}_T(t)$.

The term $\langle u(t)\rangle_{X|Y}$ has a nice connection to estimation and filtering theory: $u(t)$ is a stochastic process, of which one wants to estimate the expectation value given the observations $(Y_T(\tau)|\tau \in \mathbb{R})$ of the process $Y_T(t)$. The optimal (in the mean square error sense) estimator can be shown to be the conditional expectation value $E[u(t)|Y_T] = \langle u(t)\rangle_{X|Y_T} = F[Y_T](t)$. $F_1[.]$ given in equation (C.38) can then be regarded as a linear estimator with the kernel κ_1. If $F_1[.]$ is the optimal linear estimator, its kernel κ_1 fulfills the Wiener-Hopf equation (see (Papoulis, 1991)), which is unfortunately difficult to solve in general. It is however solvable for the following simple example: Let $z_1(t)$ and $z_2(t)$ be two stationary stochastic processes with the correlation functions:

$$\langle z_1(t+\tau)z_2(t)\rangle = \alpha\exp\left(-\frac{\tau}{\tau_0}\right) \tag{C.42}$$

$$\langle z_1(t+\tau)z_1(t)\rangle = \beta\exp\left(-\frac{\tau}{\tau_0}\right). \tag{C.43}$$

Then the optimal causal linear estimator (given by the Wiener filter) for the mean of

$z_2(t)$ given the history of $z_1(.)$ up to time t is given by:

$$\langle z_2(t)\rangle_{z_2^t|z_1^t} = E[z_2(t)|z_1(s), s \leq t] \propto \int_{\mathbb{R}} \kappa_1(t-s) z_1(s) ds. \tag{C.44}$$

Thus, in this simple example the concrete choice (C.41) of κ_1 turns out to be optimal in the mean-square error sense.

Using these results, the spike-based learning rule (C.31) becomes:

$$\frac{d}{dt} w_j(t) = \alpha \frac{Y(t)}{u(t)} \nu_j(t) \left(-\frac{u(t) - \overline{u}(t)}{\overline{u}(t)} + \beta \left(u_T(t) - \overline{u}_T(t) \right) \right) - \alpha \lambda w_j(t).$$

The modifications for the rate-based learning equation are of similar form.

C.4 A Volterra Series for the Relevance Signal Operator

In this section, a Volterra series for the operator $F[Y_T]$ is derived. The general problem is, that Y_T is rather a distribution than a uniformly bounded function, thus making the straight forward application of standard theorems (see for example (Boyd & Chua, 1985) and (Maass & Sontag, 2000)) impossible. In the following derivation, it has to be assumed that the spike train Y_T is generated by a neuron with an absolute refractory period τ_{ref}.

First, the spike trains $Y_T(t) = \sum_i \delta(t - t_T^i)$ are identified via a bijective mapping with the sequences of firing times $T \in U$, where $T = (t_T^1, t_T^2, \ldots)$. Here, $U \subseteq \mathbb{R}^{\mathbb{N}}$ is defined as the subset of all $\mathbb{R}_0^+$-valued sequences in $\mathbb{R}^{\mathbb{N}}$, which fulfill $t_{i+1} - t_i > \tau_{\text{ref}}$ and $t_i \geq 0$ for all $i \in \mathbb{N}$; thus all elements of U are ordered, positive and the sequence elements have a minimal distance τ_{ref}. The absolut refractory period τ_{ref} is a arbitrary positive number.

The operator F can be regarded as a mapping from U to the functions from $\mathbb{R}$ to $\mathbb{R}$ denoted as $\mathbb{R}^{\mathbb{R}}$. Let $C_M(\mathbb{R}, \mathbb{R})$ be the set of uniformly bounded (by a $M_1 > 0$) and uniformly Lipschitz continuous functions (with Lipschitz constant $M_2 > 0$) from $\mathbb{R}$ to $\mathbb{R}$, i. e. :

$$C_M(\mathbb{R}, \mathbb{R}) = \{ f \in \mathbb{R}^{\mathbb{R}} | \|f\| < M_1, |f(s) - f(t)| < M_2 |s - t| \}.$$

Now consider a function $\sigma \in C_M(\mathbb{R}, \mathbb{R})$ with the additional properties of its support S_σ:

$$S_\sigma \neq \emptyset, \; S_\sigma \subseteq \left[-\frac{\tau_{\text{ref}}}{2}, \frac{\tau_{\text{ref}}}{2} \right].$$

The function σ defines an injective mapping φ_σ from $T = (t_T^1, \ldots) \in U$ to $C_M(\mathbb{R}, \mathbb{R})$ via:

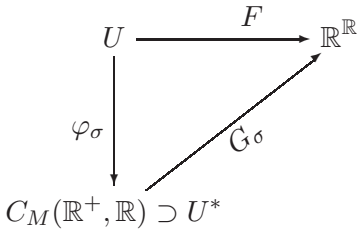
$$\varphi_\sigma(T) = \int_{\mathbb{R}} \sigma(t-s) Y_T(s) = (\sigma * Y_T)(t) = \sum_{i=0}^{\infty} \sigma(t - t_T^i).$$

The image of U under φ_σ is denoted as $U^* = \varphi_\sigma(U)$. A new operator G_σ from U^* to

$\mathbb{R}^{\mathbb{R}}$ is non-ambiguously defined for all $f \in U^*$ by:

$$G_\sigma[f] = F[\varphi_\sigma^{-1}(f)].$$

Thus one obtains the following commuting diagram:



We assume that the operator F is time-invariant and has the fading memory property, for a definition see (Boyd, Chua, & Desoer, 1984); these properties are then inherited by G_σ. Furthermore, U^* is an equicontinuous subset of all bounded and continuous functions from $\mathbb{R}$ to $\mathbb{R}$. Thus, the theorem 1 from (Boyd & Chua, 1985) can be applied, and G_σ and subsequently F can be approximated by a Volterra series:

$$F[Y_T](t) = G_\sigma[\sigma * Y_T](t) = \sum_{n=0}^{\infty} \int \cdots \int g_{n,\sigma}(t - t_1, \ldots, t - t_n) \prod_{i=1}^{n} (\sigma * Y_T)(t_i) dt_i.$$

Since convolution is associative, the operator F has the Volterra series:

$$F[Y_T](t) = \sum_{n=0}^{\infty} \int \cdots \int \kappa_n(t - t_1, \ldots, t - t_n) \prod_{i=1}^{n} Y_T(t_i) dt_i,$$

where the kernels κ_n are given by the n-fold convolution of $g_{n,\sigma}$ with σ:

$$\kappa_n(t_1, \ldots, t_n) = \int \cdots \int g_{n,\sigma}(t_1 - s_1, \ldots, t_n - s_n) \prod_{i=1}^{n} \sigma(s_i) dt_i.$$

This Volterra series can easily be shown to be independent of the choice of the function σ, since the Volterra series for the operators $G_{\sigma_1} = G_{\sigma_2}$ for $\sigma_1 \neq \sigma_2$ is unique (theorem 2.5.2 in (Boyd et al., 1984)).

C.5 Comparison with the Derivation Presented in Previous Work

In this section we want to point out the differences between the derivations given in section C.2 and the one given in the appendix A in (Klampfl et al., 2009), which lead to the different learning rules. In the first paragraph we focus on the differences between the derivations, that cause the different behavior for the continuous time limit $\Delta t \to 0$ of the learning rules. In the second paragraph other differences are mentioned shortly.

C.5.1 Differences Concerning the Continuous Time Limit

In (Klampfl et al., 2009) Klampfl et. al. start from maximizing the following objective function L_0:

$$L_0 = -I(X,Y) + \beta I(Y,Y_T) - \gamma D_{KL}(P(Y)\|P(\tilde{Y})). \tag{C.45}$$

As the important differences that are discussed in this paragraph arise from the first two terms, we may reduce the investigation on the objective function L:

$$L = \beta_1 I(X,Y) + \beta_2 I(Y,Y_T), \tag{C.46}$$

with trade-off parameters β_1 and β_2. This is exactly the same objective function as the one defined in (C.6), which served as the starting point for the derivation given in section C.2. The learning rule given in (Klampfl et al., 2009) is of the form:

$$\Delta w_j^l = \alpha \sum_{h=1}^{2} \beta_h \left\langle C_j^l M_h^l \right\rangle_{Y_T^K, Y^K, X^K}, \tag{C.47}$$

which is exactly of the same form as (C.18), which is stated here again for convenience:

$$\Delta w_j^l = \alpha \sum_{h=1}^{2} \beta_h \left\langle C_j^l \mathcal{L}_h^l \right\rangle_{Y_T^K, Y^K, X^K}. \tag{C.48}$$

C_j^l is defined by (C.20) and corresponds to C_{1j}^l in the notation of (Klampfl et al., 2009). The term M_1^l is equal to $\mathcal{L}_1^l$ defined in (C.21). The term M_2^l (corresponding to F_{12}^l in the notation of (Klampfl et al., 2009)) is given in equation (57) in (Klampfl et al., 2009), and it can be written as[4]:

$$M_2^l = y^l y_T^l \log \frac{\overline{\overline{\rho}}_{1T}^l}{\overline{\rho}_1^l \overline{\rho}_T^l} + y^l (1 - y_T^l) \log \frac{\overline{\overline{\rho}}_1^l - \overline{\overline{\rho}}_{1T}^l}{\overline{\rho}_1^l - \overline{\rho}_1^l \overline{\rho}_T^l} \tag{C.49}$$

$$+ y_T^l (1 - y^l) \log \frac{\overline{\overline{\rho}}_T^l - \overline{\overline{\rho}}_{1T}^l}{\overline{\rho}_T^l - \overline{\rho}_1^l \overline{\rho}_T^l} + (1 - y^l)(1 - y_T^l) \log \frac{1 - \overline{\overline{\rho}}_1^l - \overline{\overline{\rho}}_T^l + \overline{\overline{\rho}}_{1T}^l}{1 - \overline{\rho}_1^l - \overline{\rho}_T^l + \overline{\rho}_1^l \overline{\rho}_T^l} \tag{C.50}$$

using the following definitions, with $\rho_1^l := \rho^l$ and $Y_1 := Y$:

$$\overline{\rho}_a^l = \left\langle \rho_a^l \right\rangle_{X^l | Y_a^{l-1}}, \; a \in \{1, T\} \tag{C.51}$$

$$\overline{\overline{\rho}}_a^l = \left\langle \rho_a^l \right\rangle_{X^l | Y^{l-1}, Y_T^{l-1}} \tag{C.52}$$

$$\overline{\overline{\rho}}_{1T}^l = \left\langle \rho_1^l \rho_T^l \right\rangle_{X^l | Y^{l-1}, Y_T^{l-1}}. \tag{C.53}$$

[4]The notation used here differs from the original notation given in (Klampfl et al., 2009); here, in general, the index 1 appearing in (Klampfl et al., 2009) is left out and the index 2 is replaced by T.

The term M_2^l may be interpreted as a polynomial in the time step size Δt, where the lowest order $\mathcal{O}(\Delta t)$ is explicitly given by

$$M_2^l = y^l \log \frac{\overline{\overline{\rho}}_1^l}{\overline{\rho}_1^l} + y_T^l \log \frac{\overline{\overline{\rho}}_T^l}{\overline{\rho}_T^l} - (\overline{\overline{\rho}}_1^l + \overline{\overline{\rho}}_T^l) + (\overline{\rho}_1^l + \overline{\rho}_T^l) + \mathcal{O}(\Delta t^2) \tag{C.54}$$

However, by applying the approximation given in A.3 in (Klampfl et al., 2009):

$$\overline{\overline{\rho}}_a^l = \overline{\rho}_a^l, \tag{C.55}$$

the lowest order of the term M_2^l vanishes, and M_2^l becomes of order $\mathcal{O}(\Delta t^2)$. Thus the contribution $C_j^l M_2^l$ is strictly of higher order than the contribution $C_j^l M_1^l$ appearing in (C.47); therefore a reasonable continuous time limit of the rule presented in (Klampfl et al., 2009) does not exist, as the term stemming from the maximization of $I(Y^K, Y_T^K)$ goes to zero. The remedy, proposed in section 3.1 in (Klampfl et al., 2009), of replacing the mutual information $I(Y, Y_T)$ in L_0 by an information rate $I(Y, Y_T)/\Delta t$ does not solve this problem, as the term $I(Y, Y_T)/\Delta t$ diverges in the continuous time limit. Therefore the new objective function would ill-defined in this limit.

In the approach presented here, the term $\mathcal{L}_2^l$ is given by the following term (that is still strictly equal to the expression given in (C.49)):

$$\mathcal{L}_2^l = y^l \log \left(\frac{\left\langle \rho^l \right\rangle_{X^l|Y^{l-1},Y_T^K}}{\left\langle \rho^l \right\rangle_{X^l|Y^{l-1}}} \right) + (1 - y^l) \log \left(\frac{1 - \left\langle \rho^l \right\rangle_{X^l|Y^{l-1},Y_T^K}}{\left\langle 1 - \rho^l \right\rangle_{X^l|Y^{l-1}}} \right). \tag{C.56}$$

The term $\left\langle \rho^l \right\rangle_{X^l|Y^{l-1},Y_T^K}$ appearing in (C.56) is related to the problematic term $\overline{\overline{\rho}}^l$ via:

$$\overline{\overline{\rho}}^l = \left\langle \left\langle \rho^l \right\rangle_{X^l|Y^{l-1},Y_T^l} \right\rangle_{Y_T^{l,K}|Y^{l-1},Y_T^{l-1}}. \tag{C.57}$$

Here $Y_T^{l,K}$ is defined as $(y_T^l, \ldots, y_T^K)$. The term $\left\langle \rho^l \right\rangle_{X^l|Y^{l-1},Y_T^K}$ is approximated by a linear estimator as explained in section C.3, circumventing the problems faced in (Klampfl et al., 2009). This linear estimator could be plugged into (C.57), and the resulting term M_2^l could be approximated by its lowest order expression (C.54) in order to prevent problems in the continuous time limit of the approach taken in (Klampfl et al., 2009).

C.5.2 Other Differences

Another difference between the derivations considered here arises from the different terms D_{KL} appearing in (C.45) and the term L_3 in (C.5) in the objective functions. Both terms are introduce in order to prevent the weights from growing unboundedly. For the PCA learning rule presented in chapter 3 it is however more appropriate to use the term L_3, as the resulting learning rule exactly performs PCA, which would not have been the case if D_{KL} was chosen to enter the objective function.

Further differences between the learning rules presented in chapter 3 and the simplified learning rule given in equation (21) in (Klampfl et al., 2009) are due to the linear

neuron model and the expansion of the logarithms performed in (C.28) and (C.29) to linear order.

C.6 The Fokker-Planck Equation

The derived learning rules (C.31) and (C.33) are stochastic differential equations for the weight vector $w(t)$. In this section, the evolution of the weights is described by the means of the Fokker-Planck approach (see (Risken, 1996)): A partial differential equation (PDE) for the probability distribution of the weights is given and its peaks are quantitatively described by an ordinary differential equation. We only concentrate on the description of the dynamics of the spike-based rule (C.31); the description for the rate based rule (C.33) can be derived in a similar way and turns out to be exactly the same as for the spike-based rule.

First, the new variable $z(t) = \overline{u}(t)/\nu_0$ is introduced, which fulfills the differential equation:

$$\tau_C \frac{d}{dt} z = -z + \frac{1}{\nu_0} w(t)\nu(t), \tag{C.58}$$

where $w(t)\nu(t) = \sum_i w_i(t)\nu_i(t)$ is the standard scalar product. The state vector x is defined as:

$$x = (x_1, \ldots, x_{2N+3}) = (w_1, \ldots, w_N, \nu_1, \ldots, \nu_N, u_T, \overline{u}_T, z). \tag{C.59}$$

Now, the learning rule (C.31) is rephrased in a convenient way:

$$\frac{d}{dt} w(t) = \frac{Y(t)}{w\nu} A(x) \tag{C.60}$$

$$A(x) := \alpha\nu \left(-\frac{w\nu - \nu_0 z}{\nu_0 z} + \beta(u_T - \overline{u}_T) \right). \tag{C.61}$$

For the sake of simplicity, in this section we temporarily ignore the decay term $-\alpha\lambda w(t)$ in the rephrased learning rule (C.60); the following considerations can easily be generalized to include this drift. The temporal evolution of the joint probability density $p(x,t)$ obeys a continuity equation called the Master equation, reflection the fact that the total probability is normalized to unity at every time t:

$$\partial_t p(x,t) = \int Q(x,x',t) p(x',t) dx'. \tag{C.62}$$

The quantity $Q(x,x',t)$ is the transition rate; it is the sum of multiple contributions, which can be determined from the differential equations the variables x_i fulfill. For the spike-based learning rule (C.31), the PDE (C.62) unfortunately turns out to be nonlocal in the variable w. But for small learning rates α, it can be approximated by a local, second order PDE called the Fokker-Planck equation via a the method of Kramers-Moyal

expansion[5]. This Fokker-Planck equation reads:

$$\partial_t p(x,t) = -\sum_{i=1}^{2N+3} \partial_{x_i} \left(D_i(x) p(x,t)\right) + \frac{1}{2} \sum_{i,j=1}^{2N+3} \partial_{x_i}\partial_{x_j} \left(D_{ij}(x) p(x,t)\right). \tag{C.63}$$

The terms $D_i(x)$ are called the drift functions and the the terms $D_{ij}(x)$ are the diffusion functions. The drift and diffusion functions for the weights $w_i = x_i,\ i \leq N$ are given by:

$$D_i(x) = \frac{1}{u_0} A_i(x),\ i \leq N \tag{C.64}$$

$$D_{ij} = \frac{1}{u_0} A_i(x) A_j(x),\ i,j \leq N. \tag{C.65}$$

Furthermore, for small learning rates $\alpha \ll 1$ the variables $\nu(t)$, $u_T(t)$, $\overline{u}_T(t)$ and $z(t)$, called the fast variables, change much faster in time than the weights $w(t)$, the so-called slow variables; thus $w(t)$ may be regarded quasi-static. The marginal distribution of the fast variables settles in a equilibrium wrt. to the quasi-static value of the slow variables. Using this fact, a Fokker-Planck equation for the probability distribution $p(w,t)$ of the slow variables may be formulated by the technique of adiabatic elimination of fast variables (see (Risken, 1996) chapter 8). This PDE is given by:

$$\partial_t p(w,t) = -\sum_{i=1}^{N} \partial_{w_i} \left(D_i^0(w) p(w,t)\right) + \frac{1}{2} \sum_{i,j=1}^{N} \partial_{w_i}\partial_{w_j} \left(D_{ij}^0(w) p(w,t)\right). \tag{C.66}$$

In the lowest order approximation, the functions $D_i^0(w) =: A_i(w)$ and $D_{ij}^0(w)$ are given by:

$$D_i^0(w) =: A_i(w) = \langle D_i(x) \rangle_s \tag{C.67}$$

$$D_{ij}^0(w) =: \langle D_{ij}(x) \rangle_s \tag{C.68}$$

The average $\langle . \rangle_s$ is taken over the fast variables given static slow variables.

The drift function $A(w)$ is then given by (taking the decay term $-\alpha\lambda w$ again into account):

$$A_i(w) = \alpha \frac{1}{u_0} \left(-\frac{\sum_{j=1}^N (\langle \nu_i \nu_j \rangle - \langle \nu_i \rangle \langle \nu_j \rangle) w_j}{\sum_{j=1}^N \langle \nu_j \rangle w_j} + \beta \langle \nu_i (u_T - \overline{u}_T) \rangle \right) - \alpha\lambda w_i \tag{C.69}$$

$$= \alpha \frac{\nu_0}{u_0} \left(-\frac{\sum_{j=1}^N C_{ij} w_j}{\nu_0 \sum_{j=1}^N w_j} + \beta C^T \right) - \alpha\lambda w_i \tag{C.70}$$

$$C_{ij} = \frac{\langle \nu_i \nu_j \rangle}{\nu_0^2} - 1 \tag{C.71}$$

$$C_i^T = \frac{\langle \nu_i (u_T - \overline{u}_T) \rangle}{\nu_0} \approx \frac{\langle \nu_i u_T \rangle - \langle \nu_i \rangle \langle u_T \rangle}{\nu_0}. \tag{C.72}$$

[5] In fact, here the Kramers-Moyal expansion is an expansion in terms of the learning rate α.

In the last equation, it was assumed that the mean $\langle u_T \rangle$ is well approximated by the low-pass filter $\overline{u}_T(t)$.

One can see, that the term $A(w)$ is of first order $\mathcal{O}(\alpha)$ in the learning rate, whereas the drift D^0_{ij} is of second order $\mathcal{O}(\alpha^2)$. Thus, we expect the dynamics of the system for a small learning rate to be well described by the following deterministic differential equation:

$$\frac{d}{dt}\hat{w} = A(\hat{w}). \tag{C.73}$$

The trajectories $w(t)$ will fluctuate around the solutions $\hat{w}(t)$ of (C.73). This results in a sharp peak of the stationary distribution $p(w)$ around $\hat{w}(t)$.

It is straight-forward to show, that the eigenvalues of the matrix $DA(\hat{w})$, defined as:

$$DA_{ij} := \frac{\partial A_i(\hat{w})}{\partial \hat{w}_j}, \tag{C.74}$$

are all real and negative at the critical points w^*. Thus the critical points w^* are stable and describe the peaks of the stationary distribution $p(w)$ for small learning rate α.

Appendix D

Extended Spiking Information Bottleneck

D.1 Derivation of the Learning Rules

D.1.1 The IB Learning Rule

Here we calculate the gradient of the objective function L wrt. $\boldsymbol{w}$ and $\boldsymbol{q}$. The following relations are useful:

$$\begin{aligned}\left\langle \log F(y^0, R) \right\rangle &= \left\langle y^0 \log(F^0) + (1-y^0)\log(1-F^0) \right\rangle \\ &= \left\langle g^0 \log(F^0) + (1-g^0)\log(1-F^0) \right\rangle \\ \left\langle \log p(y^0) \right\rangle &= \left\langle g^0 \right\rangle \log \left\langle g^0 \right\rangle + (1-\left\langle g^0 \right\rangle)\log(1-\left\langle g^0 \right\rangle).\end{aligned}$$

Using these identities the objective function L can be written as:

$$L = \langle g \log F + (1-g)\log(1-F) - g \log \langle g \rangle - (1-g)\log(1-\langle g \rangle) \rangle - \frac{\gamma}{2}\boldsymbol{w}^2.$$

For the sake of simplicity the time step index was left out as no confusion can occur, e.g. $F = F^0$. From this form of L the gradient wrt. $\boldsymbol{w}$ is straight forward to calculate:

$$\begin{aligned}\frac{\partial L}{\partial \boldsymbol{w}} &= \left\langle \left(\log\left(\frac{F}{1-F}\right) - \log\left(\frac{\langle g \rangle}{1-\langle g \rangle}\right) \right) \frac{\partial g}{\partial \boldsymbol{w}} \right\rangle - \gamma \boldsymbol{w} \\ &= \left\langle \left(\sigma^{-1}(F^t) - \sigma^{-1}(\langle g \rangle) \right) \frac{\partial g}{\partial \boldsymbol{w}} \right\rangle - \gamma \boldsymbol{w}.\end{aligned}$$

Here we used the fact that the expected value $\langle \cdot \rangle$ in the above equation is only taken over the joint distribution $p(X^{-\infty}, R)$ which is independent of $\boldsymbol{w}$ (and $\boldsymbol{q}$) and hence the gradient $\frac{\partial}{\partial \boldsymbol{w}}$ (and $\frac{\partial}{\partial \boldsymbol{q}}$) commutes with the average operator $\langle \cdot \rangle$. We notice that $\log(x/(1-x))$ is the inverse function of the logistic function $\sigma(x) = 1/(1+\exp(-x))$. Further the gradient of g yields $\partial g / \partial \boldsymbol{w} = g' \boldsymbol{\nu}$, where g' is the derivative of g. This results in the $\boldsymbol{w}$-part of the learning rule (4.6).

The gradient of L wrt. $\boldsymbol{q}$ is even simpler as only F depends on $\boldsymbol{q}$. Hence only the following term has to be calculated:

$$\frac{\partial}{\partial \boldsymbol{q}} \langle g \log F + (1-g)\log(1-F) \rangle = \left\langle \frac{F'}{F(1-F)} \boldsymbol{h}(g-F) \right\rangle.$$

Using the relation $\sigma' = \sigma(1-\sigma)$, which holds for the logistic function σ, yields the final

learning rule for the parameters $\boldsymbol{q}$.

D.1.2 An InfoMax Learning Rule

In close analogy to the derivation of the IB learning rule presented above one can derive an InfoMax learning rule starting from the objective function L_{InfoMax}:

$$\begin{aligned} L_{\text{InfoMax}} &= I(y^0, X^{-\infty}) - \frac{\gamma}{2}\boldsymbol{w}^2 \\ &= \langle g \log g + (1-g)\log(1-g) - g\log\langle g\rangle - (1-g)\log(1-\langle g\rangle)\rangle - \frac{\gamma}{2}\boldsymbol{w}^2. \end{aligned}$$

The yields the following InfoMax learning rule:

$$\Delta\boldsymbol{w} = \eta_w g' \boldsymbol{\nu}\Big(\sigma^{-1}(g) - \sigma^{-1}(\langle g\rangle)\Big) - \eta_w \gamma \boldsymbol{w}.$$

D.2 Details of the Numerical Examples

D.2.1 Example of Section 4.2.2

Weights $\boldsymbol{w}$ were initialized with 0.15, the parameters $\boldsymbol{q}$ with 0 and $\hat{g}$ with 0.02. The learning rates were set to $\eta_w = 0.075$, $\eta_{q_1} = 4.25\cdot 10^{-4}$, $\eta_{q_2} = 4.25\cdot 10^{-3}$ and $\eta_g = 2\cdot 10^{-3}$. Correlated spike trains are generated using techniques described in (Gütig et al., 2003). The values of L and L_{IB} shown in Fig. 4.3**C** were estimated with the `pyentropy` software package described in (Ince, Petersen, Swan, & Panzeri, 2009) using sophisticated bias correcting methods. Every point is an average of 50 independent trials each estimated from sequences Y and R of length $5\cdot 10^5$ with frozen $\boldsymbol{w}$ and $\boldsymbol{q}$.

D.2.2 Example of Section 4.2.3

Weights $\boldsymbol{w}$ were initialized with 0.05, $\hat{g}$ with 0.02 and the initial values of the components of $\boldsymbol{q}$ were set to 0. The learning rates were set to $\eta_w = 2\cdot 10^{-3}$, $\eta_q = 10^{-3}$, $\eta_g = 2.5\cdot 10^{-3}$ and the parameters $a,\ b$ were set to $a = 1/2,\ b = 1/8$. The trade-off parameter was set to $\gamma = 6\cdot 10^{-5}$. For this example a recurrent network of $r = 200$ sigmoidal rate neurons was used (as a LSM). The state vector $\boldsymbol{s}^t = (s_1^t, \ldots, s_r^t) \in \mathbb{R}^r$ obeys the equation:

$$\boldsymbol{s}^{t+1} = \boldsymbol{s}^t \cdot (1-\alpha) + \beta f\left(W_s \boldsymbol{s}^t + W_{in}(R^t - 0.5) * 2\right),$$

with the parameters $\alpha = 0.4$ and $\beta = 0.44$. The activation function $f : \mathbb{R}^r \to \mathbb{R}^r$ is given by applying the hyperbolic tangent component-wise. The elements of the recurrent weight matrix $W_s \in \mathbb{R}^{r^2}$ are generated in the following way: The probability of two neurons to be connected was set to $1/2$, the weight for a connected pair was drawn from a normal distribution $\mathcal{N}(0,1)$. Finally W_s was rescaled by a scalar such that its spectral radius was equal to 0.8. The elements of $W_{in} = ((W_{in})_i, \ldots, (W_{in})_r) \in \mathbb{R}^r$ were drawn iid. form $\{0,1\}$ with $p((W_{in})_i = 1) = 0.3$. The filterbank h was then chosen to equal the state vector, i.e. $\boldsymbol{h}^t = \boldsymbol{s}^t$.

D.2.3 Details to the Predictive Coding Application

Weights $\boldsymbol{w}$ were initialized with 0.1 and $\hat{g}$ as well as all elements of the history of g and g' with 0.01. The learning rates were set to $\eta_w = 2 \cdot 10^{-4}$ and $\eta_g = 2.5 \cdot 10^{-4}$. The trade-off parameter was set to $\gamma = 10^{-3}$. Furthermore the values of L and $L_{\text{predictive}}$ were evaluated using the `python` module `pyentropy` based on spike trains X, Y of length $5 \cdot 10^5$ with frozen weights $\boldsymbol{w}$. For $L_{\text{predictive}}$ the term $I(y^0, X^{0,\delta})$ was approximated by $I(y^0, X^{0,\delta}) \approx \sum_{j=1}^{100} I(y^0, X_j^{0,\delta})$ to avoid the undersampling problem occurring in the evaluation of the mutual information for high-dimensional variables. This approximation introduced a large error, however the results still give intuition of the evolution of the "true" IB objective function $L_{\text{predictive}}$.

Appendix E

Computational Power and the Order-Chaos Phase Transition in Reservoir Computing

E.1 Computational Performance for Further Example Tasks

In this section the performance measure p_{exp} defined in eq. (5.1) in chapter 5 for two further example tasks is shown. In Fig. E.1 $p_{\text{exp}}(C, \text{RAND}_5)$ is plotted, averaged over 20 randomly chosen 5-bit functions RAND_5, circuits C, initial conditions and input streams. A uniform distribution over all 2^{2^5} functions $\text{RAND}_5 : \{-1, +1\}^5 \to \{-1, 1\}$ was applied. In Fig. E.2 the quantity $p_{\text{exp}}(\text{C}, \text{AND}_5)$ for the 5-bit AND-function is plotted, i.e. the target output at time t is $\text{AND}_{5,\tau}(u, t) = \max_{i\in\{1,\ldots,5\}}\{u(t - \tau - i)\}$ for delay τ. Shown results are averages over 10 circuits C, initial conditions and input streams. The results shown in Fig. E.1 and E.2 are qualitatively similar to the results on the PAR-task reported in chapter 5.

E.2 Definition and Calculation of p_∞

In this section we give a detailed definition of the heuristic performance measure p_∞ and outline an approach to calculate it efficiently.

E.2.1 Notation

Without loss of generality we may set the readout time t to 0 as the input random process $u(.)$ is assumed to be stationary. Hence the target task of the n last input bits (delay $\tau = 0$) is a function f_T of $u(-1), \ldots, u(-n)$. We introduce the following useful notation for the input:

$$\begin{aligned} u^i &:= (u^i(-1), u^i(-2), \ldots) \in \{-1, 1\}^{\mathbb{N}} \\ u^{i,t} &:= (u^i(t-1), u^i(t-2), \ldots) \in \{-1, 1\}^{\mathbb{N}} \end{aligned}$$

For a vector $x = (x_1, \ldots, x_N) \in \mathbb{R}^N$ we use the the p-norm $\|.\|_p$ with $p = 1$ (the Manhattan norm) for measuring distances:

$$\|x\|_1 = \sum_{i=1}^{N} |x_i|.$$

E.2.2 Definition of p_∞

In the following we consider two instances N1 and N2 of the same network which solely differ in the input streams $u^1(.)$ and $u^2(.)$ they receive. Let $d(k)$ be the normalized expected distance between the two network states $x^1(0)$ and $x^2(0)$ of N1 and N2 at time 0 for the case where the inputs $u^1(.)$ and $u^2(.)$ only differ at the single time step $t = -k$. Hence for given u^1, the sequence u^2 is determined by:

$$\begin{aligned} u^2(-i) &= u^1(-i) \quad \forall i \in \mathbb{N} \wedge i \neq k \\ u^2(-k) &= -u^1(-k). \end{aligned}$$

$\forall k \in \mathbb{N}$ the inputs $u^1(-k) \in \{-1, +1\}$ are iid. with probability $p(u^1(-k) = 1) =: 0.5$. $d(k)$ is then formally defined as:

$$d(k) := \frac{1}{N} \left\langle \left\langle \|x^1(0) - x^2(0)\|_1 \right\rangle_W \right\rangle_u . \tag{E.1}$$

The average $\langle . \rangle_u$ is taken over all inputs $u^1(.)$, $u^2(.)$ from the ensemble defined above and all initial conditions of the network; $\langle . \rangle_W$ denotes the average over all weight matrices W. If the network is not in the fading memory regime (or equivalently: if it does not have the echo state property, for a formal definition see (Jaeger, 2001) or (Maass et al., 2002)) $d(k)$ defined in (E.1) might well depend on the distribution of initial conditions from which the evolution of x^1 and x^2 starts; we address this subtle point below. We define p_∞ in the following way:

$$p_\infty = \max\{ \lim_{k \to \infty} (d(2) - d(k)) , 0\} = \max\{d(2) - d(\infty), 0\}.$$

E.2.3 The Annealed Approximation

We want to calculate the distance $d(k)$ defined in (E.1) for large networks $N \to \infty$ using the annealed approximation (AA) introduced in (Derrida & Pomeau, 1986). The AA consists of approximating the original dynamics of the network by assuming that the weight matrix W is drawn iid. at every time step. First we notice that in the AA and the limit $N \to \infty$ the following holds:

$$\begin{aligned} d(k) &= \left\langle \left\langle \frac{1}{N} \sum_{b=1}^{N} |x_b^1(0) - x_b^2(0)| \right\rangle_W \right\rangle_u = \left\langle \left\langle |x_a^1(0) - x_a^2(0)| \right\rangle_W \right\rangle_u \quad \forall a \in \mathbb{N} \\ d(k) &\xrightarrow{N \to \infty} \left\langle \left| \sum_{i,j=0}^{2^m - 1} p(x_a^1(t) = s_i, x_a^2(t) = s_j | u^{1,t}, u^{2,t})(s_i - s_j) \right| \right\rangle_u \quad \forall a \in \mathbb{N}. \end{aligned} \tag{E.2}$$

Here $p(x_a^1(t) = s_i, x_a^2(t) = s_j | u^{1,t}, u^{2,t})$ denotes the joint probability of finding $x_a^1(t)$ in the state s_i and $x_a^2(t)$ in the state s_j given the input $u^{1,t}$, $u^{2,t}$. Due to the AA this probability is independent of the node index a. Moreover $p(x_a^1(t) = s_i, x_a^2(t) = s_j | u^{1,t}, u^{2,t})$ determines the distribution of the recurrent feedback for the next time step. Hence, in the AA and for $N \to \infty$ the state of the network is completely described by the joint distribution of a pair of coordinates $x_a^1(t)$ and $x_a^2(t)$ in the state space $\mathcal{S} \times \mathcal{S}$. We therefore define the probability q in the following way:

$$\begin{aligned} q_{ij}(t, u^{1,t}, u^{2,t}) &:= p(x_a^1(t) = s_i, x_a^2(t) = s_j | u^{1,t}, u^{2,t}) \\ q(t, u^{1,t}, u^{2,t}) &:= \left(q_{ij}(t, u^{1,t}, u^{2,t})\right)_{i,j \in \{0,\ldots,2^m-1\}} . \end{aligned}$$

Thus $q(t, u^{1,t}, u^{2,t})$ is the $2^m \times 2^m$ matrix whose entry $q_{i,j}$ is the joint probability of finding $x_a^1(t)$ in the state s_i and $x_a^2(t)$ in s_j after applying the input $u^{1,t}$ and $u^{2,t}$ respectively. Using q we can rewrite (E.2) as:

$$\begin{aligned} d(k) &= \left\langle \left| \sum_{i,j=0}^{2^m-1} q_{ij}(0, u^1, u^2)(s_i - s_j) \right| \right\rangle_u \\ &= \left\langle \left| 2^{1-m} \sum_{i,j=0}^{2^m-1} q_{ij}(0, u^1, u^2)(i - j) \right| \right\rangle_u . \end{aligned}$$

We need to calculate $q_{ij}(t, u^{1,t}, u^{2,t})$ at time $t = 0$ in order to determine $d(k)$. This can be achieved iteratively via the mapping S representing the transition from time step t to $t + 1$ by applying the input pair $u^1(t)$ and $u^2(t)$:

$$q(t + 1, u^{1,t+1}, u^{2,t+1}) = S\left(q(t, u^{1,t}, u^{2,t}), u^1(t), u^2(t)\right) .$$

The k-fold composition of S with the inputs u^1 and u^2 is denoted by $S^{(k)}_{u^1,u^2}$:

$$\begin{aligned} q(t, u^{1,t}, u^{2,t}) &= S^{(k)}_{u^1,u^2}\left(q(t - k, u^{1,t-k}, u^{2,t-k})\right) \\ q(t, u^{1,t}, u^{2,t}) &= \lim_{k\to\infty}\left(S^{(k)}_{u^1,u^2}\left(q(t - k, u^{1,t-k}, u^{2,t-k})\right)\right) . \end{aligned}$$

For the sake of fast numerical evaluation, we calculate $q(0, u^1, u^2)$ by starting at $t = -n$ with the initial condition q^*:

$$q(0, u^1, u^2) = S^{(n)}_{u^1,u^2}(q^*) .$$

The initial condition $q^* := 2^{-m}\mathrm{id}_{2^m \times 2^m}$ (where $id_{n\times m}$ is the $n \times m$ identity matrix) was chosen because of the following relation:

$$q^* = \left\langle q(t, u^1, u^1)\right\rangle_{u^1} .$$

E.2.4 Separation approximation

Unfortunately the complexity to calculate q scales like $\mathcal{O}(2^{2m})$ (capital $\mathcal{O}$ notation) with the number of bits m. This basically renders the approach described above useless for $m > 5$. However, this can be circumvented by generalizing the AA in the following way. Since the state $x_a^i(t)$ of neuron a is quantized with m bits, we can write it in the following way:

$$\begin{aligned} x_a^i(t) &= \sum_{l=0}^{m-1} 2^{-l}(b_l^i(t) - 1/2), \quad b_l^i(t) \in \{0,1\} \\ B_l(x_a^i(t)) &:= b_l^i(t). \end{aligned} \tag{E.3}$$

According to (E.3) there is a unique binary representation of $x_a^i(t)$ given by $(b_0^i(t), \ldots, b_{m-1}^i(t))$; the mapping $B_l(.)$ maps a state to its l^{th} bit. Equation (E.3) can be interpreted as effectively replacing unit a with m binary units of states $b_l^i(t)$ whose outputs are reduced by $1/2$ and multiplied by 2^{-l} and finally summed up; this is still exact. Now we assume that each of these m units receives input drawn independently from the input distribution and has different weights drawn independently from $N(0, \sigma^2)$ every time step; hence this approach generalizes the AA. Therefore, for given presynaptic input the $b_l^i(t)$ are independent for different i and l under this approximation. Thus:

$$\begin{aligned} q_{ij}(t, u^{1,t}, u^{2,t}) &\approx \prod_{l=0}^{m-1} q^l_{B_l(s_i),B_l(s_j)}(t, u^{1,t}, u^{2,t}) \\ q^l(t, u^{1,t}, u^{2,t}) &= (q^l_{b_l^1,b_l^2}(t, u^{1,t}, u^{2,t}))_{b_l^1,b_l^2 \in \{0,1\}}. \end{aligned}$$

$q^l(t, u^{1,t}, u^{2,t})$ is the 2×2 matrix, whose entry $q^l_{b_l^1,b_l^2}(t, u^{1,t}, u^{2,t})$ is the joint probability of finding the bit number l of unit $x_a^1(t)$ in state $b_l^1 \in \{0,1\}$ and of unit $x_a^2(t)$ in state $b_l^2 \in \{0,1\}$. Under this approximation we only have to calculate $4m$ matrix entries instead of the 2^{2m} entries, which is a considerable reduction of complexity.
We denote the update mapping for q^l by S^l:

$$q^l(t+1, u^{1,t+1}, u^{2,t+1}) = S^l(q(t, u^{1,t}, u^{2,t}), u^1(t), u^2(t)).$$

An explicit form for S^l can be derived in the following way. First we condition the probability $q^l(t+1, u^{1,t+1}, u^{2,t+1})$ on the presynaptic input $h^1 = (h_1^1, \ldots, h_K^1)$ for network N1 and $h^2 = (h_1^2, \ldots, h_K^2)$ for network N2 and on the weight matrix W for this time step (which is the same for N1 and N2). The pairs $(h_i^1, h_i^2),\ i \in 1, \ldots, K$ are iid. according to $q(t, u^{1,t}, u^{2,t})$. This yields:

$$\begin{aligned} q^l(t+1, u^{1,t+1}, u^{2,t+1}) &= \left\langle \left\langle q^l(t+1, u^{1,t+1}, u^{2,t+1} | h^1, h^2, w) \right\rangle_W \right\rangle_{h^1,h^2} \\ &= \sum_{h^1,h^2} p(h^1, h^2) \int dW\ p(W) q^l(t+1, u^{1,t+1}, u^{2,t+1} | h^1, h^2, W) \end{aligned} \tag{E.4}$$

We used the following abbreviations :

$$\begin{aligned}
\left\langle X(h^1,h^2)\right\rangle_{h^1,h^2} &:= \sum_{h^1,h^2} p(h^1,h^2)X(h^1,h^2) \\
&= \sum_{h_1^1,h_1^2}\cdots\sum_{h_K^1,h_K^2}\left(\prod_{i=1}^{K} q_{h_i^1,h_i^2}(t,u^{1,t},u^{2,t})\right)X(h^1,h^2) \qquad \text{(E.5)} \\
\langle X(W)\rangle_W &:= \int dW\ p(W)X(W) = \int_{\mathbb{R}}\cdots\int_{\mathbb{R}} X(W)p(w_1,\ldots,w_K)\ dw_1\cdots dw_K.
\end{aligned}$$

Here $w_1,\ldots,w_K$ denote the K presynaptic weights. Conditioned on the input and the weights, the network realizations N1 and N2 are independent:

$$\begin{aligned}
q_{ij}^l(t+1,u^{1,t+1},u^{2,t+1}|h^1,h^2,w) &= p(b_l^1(t+1)=i|h^1,w)p(b_l^2(t+1)=j|h^2,w) \\
&= \delta(B_l(f_m(w^Th^1+u^1(t))),i)\delta(B_l(f_m(w^Th^2+u^2(t))),j) \\
w^Th^i &:= \sum_{\alpha=1}^{K} w_\alpha h_\alpha^i.
\end{aligned}$$

$\delta(.,.)$ denotes the Kronecker-Delta of its two arguments. The K-fold integral over the weights appearing in (E.4) can be reduced to a single integral:

$$\int dW\ p(W)q_{ij}^l(t+1,u^{1,t+1},u^{2,t+1}|h^1,h^2,W) = F_{ij}^l(h^1,h^2,u^1(t),u^2(t))$$

$$\begin{aligned}
F_{ij}^l(h^1,h^2,u^1(t),u^2(t)) = \frac{1}{(8\pi\det(C)\Sigma_{22})^{1/2}}\sum_{\alpha=0}^{2^l-1}\int_{I_{\alpha,i}-u^1(t)} dv\ \exp\left(-\frac{v^2}{2}\left(\Sigma_{11}-\frac{\Sigma_{12}^2}{\Sigma_{22}}\right)\right)\times \\
\sum_{\beta=0}^{2^l-1}\left[\mathrm{erf}\left(\frac{(I_{\beta,j}^+-u^2(t))\Sigma_{22}+\Sigma_{21}v}{(2\Sigma_{22})^{1/2}}\right)-\ \mathrm{erf}\left(\frac{(I_{\beta,j}^--u^2(t))\Sigma_{22}+\Sigma_{21}v}{(2\Sigma_{22})^{1/2}}\right)\right]
\end{aligned}$$

Here we use the following definitions:

$$\begin{aligned}
C &= \sigma^2\begin{pmatrix}(h^1)^Th^1 & (h^1)^Th^2\\ (h^2)^Th^1 & (h^2)^Th^2\end{pmatrix} \\
\Sigma_{ij} &= \left(C^{-1}\right)_{ij} \\
\bigcup_{\alpha=0}^{2^l-1} I_{\alpha,i} &= \mathrm{supp}(\delta(B_l(f_m(.)),i)) \\
I_{\alpha,i} &= [I_{\alpha,i}^-,I_{\alpha,i}^+] \\
I_{\alpha,i}^- &:= \tanh^{-1}\left(2^{-l+1}\alpha-1\right) \\
I_{\alpha,i}^+ &:= \tanh^{-1}\left(2^{-l+1}(\alpha+1)-1\right).
\end{aligned}$$

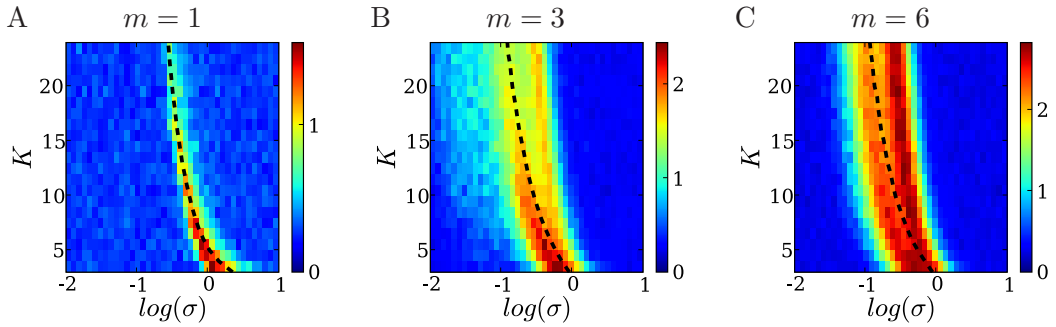


Figure E.1: The performance $p_{\text{exp}}(C, \text{RAND}_5)$ for three different quantization levels $m = 1, 3, 6$, averaged over 20 randomly drawn functions of 5 bits, circuits C, initial conditions and input streams. $p_{\text{exp}}(C, \text{RAND}_5)$ is plotted as a function of the network in-degree K and the weight STD σ. The networks size is $N = 150$. The input time series have length 10000. The solid line represents the numerically found critical line.

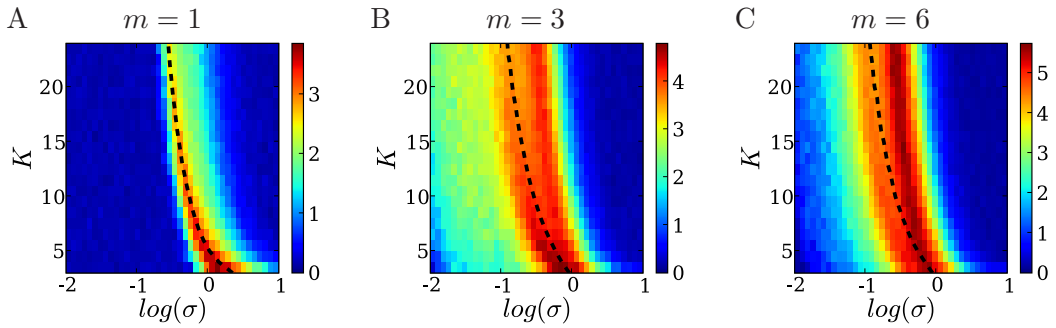


Figure E.2: Same figure as Fig. E.1 showing the performance $p_{\text{exp}}(C, \text{AND}_5)$ for the 5-bit AND-task averaged over 10 circuits C, initial conditions and input streams.

The support of the function $\delta(B_l(f_m(.)), i)$ is the union of 2^l disjoint intervals $I_{\alpha,i}$ with lower bound $I^-_{\alpha,i}$ and upper bound $I^+_{\alpha,i}$.
Using the expression F^l_{ij} the update can finally be written as:

$$q^l_{ij}(t+1, u^{1,t+1}, u^{2,t+1}) \quad = \quad \left\langle F^l_{ij}(h^1, h^2, u^1(t), u^2(t)) \right\rangle_{h^1,h^2} .$$

For the sake of computational tractability for larger m and K, we do not evaluate the $2K$ sums explicitly involved in the average over the presynaptic input $\langle . \rangle_{h^1,h^2}$. Instead we determine this expectation value by a finite number a samples from the joint distribution $p(h^1, h^2)$; this sampling is easily realizable since $p(h^1, h^2)$ is of the product form given in (E.5); sample size for all experiments was chosen to be 150.

Appendix F

Connectivity, Dynamics, and Memory in Reservoir Computing with Binary and Analog Neurons

F.1 Lyapunov Exponents via Branching Processes

In this section the probabilities $p_{ij}^{\alpha\beta}$ defining the multitype branching process described in section 6.2 are calculated. The network is assumed to be of infinite size ($N = \infty$) and the weight matrix W is assumed to be regenerated independently at every time step (AA approximation). In the AA the recurrent input $\sum_{j=1}^{\infty} w_{ij}x_j(t)$ to unit i at time t is distributed symmetrically wrt. 0 as the weights w_{ij} are distributed symmetrically wrt. 0. It is straight forward to see that all dynamics are invariant whether the input $u(t) = 1$ or $u(t) = -1$. Hence without loss of generality (wlog) we can assume $u(t) = 1$ in the rest of the analysis, a fact that makes the use of branching theory possible.

In order to calculate the probabilities $p_{ij}^{\alpha\beta}$ it is necessary to calculate the steady-state probabilities $p_{\mathcal{S}_m}(s)$ of a network unit to assume the state $s \in \mathcal{S}_m$. Let $z_{ij}(t)$ denote the product $w_{ij}x_j(t)$ of the weight w_{ij} (with $w_{ij} \sim p_w = \mathcal{N}(0, \sigma^2)$) and the activation $x_j(t)$ of unit j at time t. All $z_{ij}(t)$ for $i, j \in \mathbb{N}$ are iid. according to p_z. The recurrent input $Z_i(t) = \sum_{j=1} z_{ij}(t)$ to unit i is the sum of K independent contributions (corresponding to the K recurrent connections for each unit). The following equations hold for the distributions:

$$\begin{aligned} p_z(z) &= \int ds \frac{1}{|s|} p_{\mathcal{S}_m}(s) p_w(z/s) \\ p_Z &= *^K p_z = \underbrace{p_z * \ldots * p_z}_{K\text{-times}} \\ p_{\mathcal{S}_m}(s) &= \int dZ p_Z(Z) \chi_{I_s}(Z+1), \end{aligned} \tag{F.1}$$

where $*^K$ denotes the K-fold convolution and $\chi_I(\cdot)$ is the indicator function of the interval I. The interval $I_s = (\psi_m \circ g)^{-1}(s)$ is the inverse image of s under the quantized activation function $\psi_m \circ g$. The equations (F.1) are solved numerically in an iterated fashion for the steady-state distributions $p_{\mathcal{S}_m}$ and p_z.

$p_{ij}^{\alpha\beta}$ denoting the probability per output link that a perturbation $s_\alpha \to s_\beta$ causes a perturbation $s_i \to s_j$ (where $\mathcal{S}_m = \{s_1, \ldots, s_{2^m}\}$ is the single unit state space) can easily

be calculated using the steady-state distributions $p_{\mathcal{S}_m}$ and p_z from (F.1). It is given by:

$$\begin{aligned} p_{ij}^{\alpha\beta} &= \int dw \int dZ'\ p_w(w) p_{Z'}(Z') \chi_{I_{s_i}}(Z' + s_\alpha w + 1) \chi_{I_{s_j}}(Z' + s_\beta w + 1) \\ p_{Z'} &= *^{K-1} p_z \end{aligned}$$

where w denotes the weight of the input that is perturbed by the $s_\alpha \to s_\beta$ perturbation and Z' denotes the recurrent input to the unit from the remaining $K-1$ presynaptic units which are unaffected by the $s_\alpha \to s_\beta$ perturbation.

In an infinite size network with neuron in-degree K generated as described in section 6.2 the neuron out-degree is Poisson distributed with mean K. Therefore the mean descendant matrix M whose element $M_{\alpha+2^m(\beta-1),i+2^m(j-1)}$ denotes the average number of descendants of type $s_i \to s_j$ caused by a $s_\alpha \to s_\beta$ perturbation for $\alpha, \beta, i, j = 1, \ldots, m$ is given by:

$$M_{\alpha+2^m(\beta-1),i+2^m(j-1)} = K \cdot p_{ij}^{\alpha\beta}.$$

According to (6.2) the largest Lyapunov exponent is defined as:

$$\begin{aligned} \lambda &= \lim_{n\to\infty} \frac{1}{n} \ln\left(\frac{H(n)}{H(0)}\right) \\ H(n) &= d(\mathbf{x}^1(n), \mathbf{x}^2(n)) = \|\mathbf{x}^1(n) - \mathbf{x}^2(n)\|_1. \end{aligned}$$

Here we restricted ourselves wlog to the use of the $p-1$ norm (as all norm are equivalent on $\mathbb{R}^N$). Following (Athreya & Ney, 1972) the expected number of perturbations after n time steps is given by $Z^T M^n$. Here Z^T denotes the transposed of $Z \in \mathbb{N}^{2^{2m}}$ whose i-th element Z_i denotes the initial number of perturbations of type $s_\alpha \to s_\beta$ with $i = \alpha + 2^m(\beta - 1)$ at time step 0. Hence $H(n)$ can be cast into the following form:

$$H(n) = Z^T M^n D$$

where the elements of $D \in \mathbb{R}^{2^{2m}}$ are given by $D_{\alpha+2^m(\beta-1)} = |s_\alpha - s_\beta|$. It is then straightforward to see that:

$$\lambda = \lim_{n\to\infty} \frac{1}{n} \ln\left(\frac{Z^T M^n D}{Z^T D}\right) = \ln(\rho)$$

where ρ is the largest eigenvalue of M which is guaranteed to be non-negative by the Perron–Frobenius theorem.

Although the dimension of M is $2^{2m} \times 2^{2m}$ we cannot expect to obtain 2^{2m} meaningful eigenvalues of M as the matrix also contains 2^m rows describing the "diagonal" perturbations $s_\alpha \to s_\alpha$ which are only trivial perturbations. Furthermore as the network dynamics of a qESN defined in (6.2) are symmetric wrt. 0, the perturbations $s_\alpha \to s_\beta$ are equivalent to the perturbation $s_{2^m+1-\alpha} \to s_{2^m+1-\beta}$. Hence, M only has $2^{m-1}(2^m - 1)$ meaningful eigenvalues of which the logarithm represents the Lyapunov spectrum.

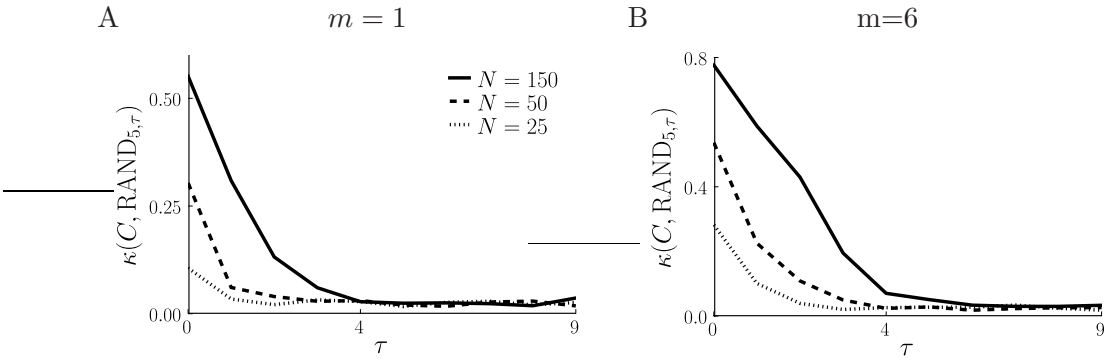


Figure F.1: Cohen's kappa coefficient $\kappa(C, \mathrm{RAND}_{5,\tau})$ plotted as a function of the delay parameter τ for binary networks with $m = 1$ ($\log \sigma = 0.2$, $K = 3$, panel A) and for high-resolution networks with $m = 6$ ($\log \sigma = 0$, $K = 3$, panel B). The plots show results for different networks with size $N = 25$ (dotted), $N = 50$ (dashed) and $N = 150$ (solid). Results for each parameter pair (τ,N) were averaged over 50 different circuits C. The performance measure $p_{\mathrm{exp}}(C, \mathrm{TASK}_n)$ can be interpreted as the area under the plotted curves (corresponding to a summation over τ).

F.2 Dependence of Computational Performance on Task Delay and Network Size

The performance measure $p_{\mathrm{exp}}(C, \mathrm{TASK}_n)$ introduced in (6.3) for a task TASK_n of n bits (e. g. PAR_5) and a circuit C is given by summation of the performances $\kappa(C, \mathrm{TASK}_{n,\tau})$ for the τ-delayed task $\mathrm{TASK}_{n,\tau}$ over all delays $\tau \geq 0$. The quantity $p_{\mathrm{exp}}(C, \mathrm{TASK}_n)$ is hence not easy to interpret as it is not bounded for all system sizes $N \in \mathbb{N}$. To give some intuition about the performance of the qESN, the kappa coefficient $\kappa(C, \mathrm{RAND}_{n,\tau})$ (which is in $[0, 1]$) is explicitly plotted as a function of τ in Fig. F.1 A for binary ($m = 1$) and in Fig. F.1 B for high-resolution networks ($m = 6$) with different network sizes $N = 25, 50, 150$ for the task RAND_5. It can be observed that for fixed N the kappa coefficient κ decreases monotonically with τ and for fixed τ it increases monotonically with N (in agreement with the results presented in Fig. 6.10). The performance measure $p_{\mathrm{exp}}(C, \mathrm{TASK}_n)$ is the area under the curves shown in Fig. F.1.

F.3 Input Separation $d(k)$ in the Annealed Approximation

Here we calculate the distance $d(k)$ defined in (6.4) for large networks $N \to \infty$ using the annealed approximation (AA) introduced in (Derrida & Pomeau, 1986). We denote networks receiving the the input $u^1(\cdot)$ and $u^2(\cdot)$ with N1 and N2 respectively. First we

notice that in the AA and the limit $N \to \infty$ the following holds:

$$\begin{aligned} d(k) &= \lim_{N\to\infty} \left\langle \frac{1}{N} \sum_{b=1}^{N} |x_b^1(0) - x_b^2(0)| \right\rangle_C = \left\langle |x_a^1(0) - x_a^2(0)| \right\rangle_C \quad \forall a \in \mathbb{N} \\ &= \sum_{i,j=0}^{2^m-1} q_{ij}(k, u^1, u^2) \, |s_i - s_j| \, . \end{aligned} \tag{F.2}$$

Here $q_{ij}(k, u^1, u^2)$ denotes the joint probability of finding $x_a^1(k)$ in the state s_i and $x_a^2(k)$ in the state s_j given the input $u^1(\cdot)$, $u^2(\cdot)$. Due to the AA this probability is independent of the node index a. Hence, in the AA and for $N \to \infty$ the state of the network is completely described by the joint distribution of a pair of states $x_a^1(k)$ and $x_a^2(k)$ in the state space $\mathcal{S}_m \times \mathcal{S}_m$. Moreover $q_{ij}(k, u^1, u^2)$ determines the distribution of the recurrent feedback for the next time step $k+1$. We define the matrix $q(k, u^1, u^2)$ with the entries $q_{ij}(k, u^1, u^2)$. We denote the mapping from $q_{ij}(k, u^1, u^2)$ to $q_{ij}(k+1, u^1, u^2)$ as S representing the transition from time step k to $k+1$ by applying the input pair $u^1(k)$ and $u^2(k)$:

$$q(k+1, u^1, u^2) = S\left(q(k, u^1, u^2)\right).$$

Separation Approximation

Since the state $x_a^i(k)$ of neuron a is quantized with m bits, we can write it in the following way:

$$\begin{aligned} x_a^i(k) &= \sum_{l=0}^{m-1} 2^{-l}(b_l^i - 1/2), \quad b_l^i \in \{0, 1\} \\ B_l(x_a^i(k)) &:= b_l^i. \end{aligned} \tag{F.3}$$

According to (F.3) there is a unique binary representation of $x_a^i(k)$ given by $(b_0^i, \ldots, b_{m-1}^i)$; the mapping $B_l(.)$ maps a state to its l^{th} bit. Equation (F.3) can be interpreted as effectively replacing unit a with m binary units of states b_l^i whose outputs are reduced by $1/2$ and multiplied by 2^{-l} and finally summed up; this is still exact. Now we assume that each of these m units receives input drawn independently from the input distribution and has different weights drawn independently from $N(0, \sigma^2)$ every time step. For given presynaptic input the b_l^1 and b_l^2 are independent for all $l = 0, \ldots, m-1$ under this approximation:

$$q_{ij}(k, u^1, u^2) = \prod_{l=0}^{m-1} q^l_{B_l(s_i), B_l(s_j)}(k, u^1, u^2),$$

$q^l(k, u^1, u^2)$ denotes the 2×2 matrix, whose entry $q^l_{b_l^1, b_l^2}(k, u^1, u^2)$ is the joint probability of finding the bit number l of unit $x_a^1(k)$ in state $b_l^1 \in \{0, 1\}$ and of unit $x_a^2(k)$ in state $b_l^2 \in \{0, 1\}$. Under this approximation we only have to calculate $4m$ matrix entries

instead of the 2^{2m} entries, which is a considerable reduction of complexity.

We denote the update mapping for q^l by S^l:

$$q^l(k+1,u^1,u^2) = S^l(q(k,u^1,u^2)).$$

An explicit form for S^l can be derived in the following way: First we condition the probability $q^l(k+1,u^1,u^2)$ on the presynaptic input $h^1 = (h^1_1,\ldots,h^1_K) \in \mathcal{S}^K_m$ for network N1 and $h^2 = (h^2_1,\ldots,h^2_K) \in \mathcal{S}^K_m$ for network N2 and on the weight matrix W defining the circuit C for this time step $k+1$ (which is the same for N1 and N2). The pairs $(h^1_i, h^2_i),\ i \in 1,\ldots,K$ are iid. according to $q(k,u^1,u^2)$. This yields:

$$\begin{aligned} q^l(k+1,u^1,u^2) &= \left\langle \left\langle q^l(k+1,u^1,u^2|h^1,h^2,W)\right\rangle_C \right\rangle_{h^1,h^2} \\ &= \sum_{h^1,h^2} p(h^1,h^2) \int dW\ p(W) q^l(k+1,u^1,u^2|h^1,h^2,W). \quad \text{(F.4)} \end{aligned}$$

Here $p(W)$ denotes the probability of the weight matrix W, which is multi-normal for all the non-vanishing K weights per row. Conditioned on the input and the weights, the network realizations N1 and N2 are independent and the K-fold integral over the weights appearing in (F.4) can be explicitly integrated to a single integral:

$$\int dW\ p(W) q^l_{ij}(k+1,u^1,u^2|h^1,h^2,W) = F^l_{ij}(h^1,h^2,u^1(k),u^2(k)).$$

F^l is defined as:

$$F^l_{ij}(h^1,h^2,u^1(k),u^2(k)) = \frac{1}{(8\pi \det(C)\Sigma_{22})^{1/2}} \sum_{\alpha=0}^{2^l-1} \int_{I_{\alpha,i}-u^1(k)} dv\ \exp\left(\frac{v^2}{2}\left(\frac{\Sigma_{12}^2}{\Sigma_{22}} - \Sigma_{11}\right)\right) \times$$
$$\sum_{\beta=0}^{2^l-1} \left[\mathrm{erf}\left(\frac{(I^+_{\beta,j} - u^2(k))\Sigma_{22} + \Sigma_{21} v}{(2\Sigma_{22})^{1/2}}\right) - \mathrm{erf}\left(\frac{(I^-_{\beta,j} - u^2(k))\Sigma_{22} + \Sigma_{21} v}{(2\Sigma_{22})^{1/2}}\right)\right].$$

Here we use the following definitions:

$$\begin{aligned} R &= \sigma^2 \begin{pmatrix} (h^1)^T h^1 & (h^1)^T h^2 \\ (h^2)^T h^1 & (h^2)^T h^2 \end{pmatrix} \\ \Sigma_{ij} &= \left(R^{-1}\right)_{ij} \\ I_{\alpha,i} &= [I^-_{\alpha,i}, I^+_{\alpha,i}] := \left[\tanh^{-1}\left(2^{-l+1}\alpha - 1\right), \tanh^{-1}\left(2^{-l+1}(\alpha+1) - 1\right)\right], \end{aligned}$$

where $(h^a)^T h^b = \sum_{i=1}^N h^a_i h^b_i$ denotes the standard dot product. Using the above expression for F^l_{ij} the update can finally be written as:

$$q^l_{ij}(k+1,u^1,u^2) = \left\langle F^l_{ij}(h^1,h^2,u^1(k),u^2(k))\right\rangle_{h^1,h^2}.$$

For the sake of computational tractability for larger m and K, we do not evaluate the $2K$ sums explicitly involved in the average over the presynaptic input $\langle . \rangle_{h^1,h^2}$. Instead we determine this expectation value by a finite number a samples from the joint distribution $p(h^1, h^2)$; this sampling is easily realizable since $p(h^1, h^2)$ is of the product form given in (F.4); sample size for all experiments was chosen to be 150.

F.4 Bound for the Memory Function

F.4.1 Upper Bound for the Memory Function

Here we derive the upper bound (6.9) for the memory function $m(k)$. The target output at time t for the memory task of k time steps is given by $y_T(t) = u(t-k)$. The linear readout with weights α_i has the output $y(t) = \alpha^T x(t) = \sum_{i=1}^N \alpha_i x_i(t)$. α is the learned by linear regression yielding $\alpha = A^{-1}p(k)$, where $A = \left\langle x(t)x(t)^T \right\rangle$ is the covariance matrix of the network state $x(t)$ and $p_k = (p_{k1}, \ldots, p_{kN}) = \langle x(t)y_T(t) \rangle$. Here $\langle \cdot \rangle$ denotes the average over the input $u(\cdot)$ for a given circuit A. Using the expression for the memory function given in (White et al., 2004) results in:

$$m(k) = p_k^T A^{-1} p_k \leq \|A^{-1}\|_2 \|p_k\|^2,$$

where $\| \cdot \|$ is the standard Euclidean norm and $\| \cdot \|_2$ is the corresponding induced operator norm. Without loss of generality we can set the readout time $t = 0$ as the input sequence $u(\cdot)$ is stationary. Now look at a single component of p_k:

$$p_{ki} = \langle x_i[u](0) u_k \rangle .$$

Here $x[u](0) = (x_1[u](0), \ldots, x_N[u](0))$ is the state vector that results from applying the left-infinite input sequence $u = (u_1, u_2, \ldots)$ where $u_k := u(-k)$. Further we define:

$$\begin{aligned} u^i &:= (u_{i+1}, u_{i+2}, \ldots) \\ u^{i,j} &:= (u_{i+1}, \ldots, u_j). \end{aligned}$$

The $\circ$ is used for concatenating sequences such that $u = u^{1,k} \circ u^k = u^{1,k-1} \circ u_k \circ u^k$ $\forall k \in \mathbb{N}$. Using this notation the component p_{ki} can be written as:

$$\begin{aligned} p_{ki} &= \langle x_i[u](0) u_k \rangle = \sum_u p(u) x_i[u](0) u_k \\ &= \sum_{u^{0,k-1}} p(u^{0,k-1}) \sum_{u_k} p(u_k) \sum_{u^k} p(u^k) \left(x_i[u^{0,k-1} \circ u_k \circ u^k](0) u_k \right). \end{aligned}$$

Since all inputs are independent, we can carry out explicitly the sum over u_k with $p(u_k) = 1/2$:

$$\begin{aligned} p_{ki} &= \frac{1}{2} \sum_{u^{0,k-1}} p(u^{0,k-1}) \sum_{u^k} p(u^k) \left(x_i[u^{0,k-1} \circ (+1) \circ u^k](0) - x_i[u^{0,k-1} \circ (-1) \circ u^k](0) \right) \\ &= \frac{1}{2} \left\langle x_i[u^{0,k-1} \circ (+1) \circ u^k](0) - x_i[u^{0,k-1} \circ (-1) \circ u^k](0) \right\rangle_{u^{0,k-1}, u^k}. \end{aligned}$$

Now the vector $h = (h_1, \ldots, h_N)$ is defined in the following way:

$$h_i := x_i[u^{0,k-1} \circ (+1) \circ u^k](0) - x_i[u^{0,k-1} \circ (-1) \circ u^k](0).$$

It can be seen that from the definition (6.4) of $d(k)$ that the following identity holds:

$$d(k) = \frac{1}{N} \langle \|h\|_1 \rangle .$$

The squared Euclidean norm of p_k can be expressed as:

$$\begin{aligned} \|p_k\|^2 &= \frac{1}{4} \sum_i^N \langle h_i \rangle^2 \frac{1}{4} \leq \frac{1}{4} \sum_i^N \langle |h_i| \rangle^2 \\ &\leq \frac{1}{4} \left(\sum_i^N \langle |h_i| \rangle \right)^2 = \frac{1}{4} \left(\left\langle \sum_i^N |h_i| \right\rangle \right)^2 = \frac{1}{4} \langle \|h\|_1 \rangle^2 = \frac{N^2}{4} d(k)^2. \end{aligned}$$

Together with the fact that $m(k) \leq 1$ (which is simply the Cauchy-Schwarz inequality) this finally results in:

$$m(k) \leq \min \left\{ \frac{N^2}{4} \|A^{-1}\|_2 \, d(k)^2, 1 \right\}.$$

F.4.2 An Annealed Approximation for $\|A^{-1}\|_2$

Here we calculate $\|A^{-1}\|_2$ for $m = 1$ (binary) qESNs using the annealed approximation (AA). We start by explicitly calculating the components $A_{ij} = \langle x_i(t) x_j(t) \rangle$. The diagonal elements simply evaluate to:

$$A_{ii} = \langle x_i(t) x_i(t) \rangle = \left\langle x_i(t)^2 \right\rangle = 0.25 =: a.$$

The network state update equations are rewritten in the following form:

$$\begin{aligned} x_i(t) &= \Theta \left(\frac{1}{2} z_i(t-1) + u(t-1) \right) \\ z_i(t-1) &:= 2 \sum_{j=1}^K w_{ij} x_j(t-1), \end{aligned}$$

where $\Theta(x) = 1/2$ for $x > 0$ and $\Theta(x) = -1/2$ otherwise. Independent from the input, the quantity $z_i(t-1)$ is normally distributed according to $z_i(t) \sim \mathcal{N}(0, K\sigma^2)$ and $z_i(t-1)$ and $z_j(t-1)$ are iid for $i \neq j$. Further, as $\langle z_i(t-1)\rangle = 0$ we may assume wlog $u(t-1) = 1$. The probability p_+ for $x_i(t) = +0.5$ and p_- evaluate to:

$$\begin{aligned} p(x_i(t) = +0.5) = p_+ &= \Phi\left(\frac{2}{K^{1/2}\sigma}\right) \\ p(x_i(t) = -0.5) = p_- &= \Phi\left(-\frac{2}{K^{1/2}\sigma}\right) \\ \Phi(x) &= \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right). \end{aligned}$$

Now the off-diagonal elements of A can easily be computed ($i \neq j$):

$$A_{ij} = \langle x_i(t)x_j(t)\rangle = 0.25(p_+ - p_-)^2 := b.$$

Due to the simple form of A (all diagonal elements are equal to a and the off-diagonal elements are equal to b) its eigenvalues can easily be determined. There are two different eigenvalues $\lambda_{\min} < \lambda_{\max}$:

$$\begin{aligned} \lambda_{\max} &= a + (N-1)b \\ \lambda_{\min} &= a - b = 0.25\left(1 - (p_+ - p_-)^2\right) \\ &= 0.25\left(1 - \left(\Phi\left(\frac{2}{K^{1/2}\sigma}\right) - \Phi\left(-\frac{2}{K^{1/2}\sigma}\right)\right)^2\right). \end{aligned}$$

Now the following relation holds for the matrix norm $\|\cdot\|_2$:

$$\|A^{-1}\|_2 = \lambda_{\min}^{-1} = 4\left(1 - \left(\Phi\left(\frac{2}{K^{1/2}\sigma}\right) - \Phi\left(-\frac{2}{K^{1/2}\sigma}\right)\right)^2\right)^{-1}.$$

References

Artola, A., Bröcher, S., & Singer, W. (1990, September). Different voltage-dependent thresholds for inducing long-term depression and long-term potentiation in slices of rat visual cortex. *Nature*, *347*, 69-72.

Athreya, K. B., & Ney, P. E. (1972). *Branching Processes*. Springer.

Badel, L., Lefort, S., Brette, R., Petersen, C., Gerstner, W., & Richardson, M. (2008). Dynamic I-V curves are reliable predictors of naturalistic pyramidal-neuron voltage traces. *J Neurophysiol*, *99*, 656 - 666.

Barrett, A., Billings, G., Morris, R., & Rossum, M. van. (2009). State based model of long-term potentiation and synaptic tagging and capture. *Plos Comp Biol*, *5(1)*, e1000259. doi:10.1371/journal.pcbi.1000259.

Becker, S. (1996, January). Mutual information maximization: models of cortical self-organization. *Network: Computation in Neural Systems*, *7*(1), 7-31.

Becker, S., & Hinton, G. E. (1992). Self-organizing neural network that discovers surfaces in random-dot stereograms. *Nature*, *355*, 161-163.

Beggs, J. M., & Plenz, D. (2003). Neuronal avalanches in neocortical circuits. *The Journal of Neuroscience*, *23*(35), 11167–11177.

Bell, A. J., & Sejnowski, T. J. (1995). An Information-Maximization Approach to Blind Separation and Blind Deconvolution. *Neural Computation*, *7*(6), 1129-1159.

Bertschinger, N., & Natschlaeger, T. (2004). Real-time computation at the edge of chaos in recurrent neural networks. *Neural Computation*, *16*(7), 1413–1436.

Bi, G., & Poo, M. (1998). Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. *J Neuroscience*, *18*(24), 10464–10472.

Bi, G., & Poo, M. (2001). Synaptic modification of correlated activity: Hebb's postulate revisited. *Ann. Rev. Neurosci.*, *24*, 139-166.

Bialek, W., Steveninck, R. R. de Ruyter van, & Tishby, N. (2006). Efficient representation as a design principle for neural coding and computation. In *IEEE International Symposium on Information Theory.* .

Blais, B., Shouval, H., & Cooper, L. (1998). Receptive field formation in natural scene environments: comparison of single-cell learning rules. *Neural Computation*, *10*, 1797-1813.

Boyd, S., & Chua, L. O. (1985). Fading memory and the problem of approximating nonlinear oparators with Volterra series. *IEEE Trans. on Circuits and Systems*, *32*, 1150–1161.

Boyd, S., Chua, L. O., & Desoer, C. A. (1984). Analytical Foundations of Volterra Series. *IMA Journal of Mathematical Control & Information*, *1*, 243-282.

Brader, J., Senn, W., & Fusi, S. (2007). Learning real-world stimuli in a neural network with spike-driven synaptic dynamics. *Neural Computation*, *19*, 2881-2912.

Brette, R., & Gerstner, W. (2005). Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *Neurophysiol*, *94*, 3637–3642.

Buonomano, D., & Maass, W. (2009). State-dependent computations: Spatiotemporal processing in cortical networks. *Nature Reviews in Neuroscience*, *10*(2), 113–125.

Buonomano, D., & Merzenich, M. (1998). Cortical plasticity: From synapses to maps. *Annual Review of Neuroscience*, *21*, 149-186.

Caporale, N., & Dan, Y. (2008). Spike Timing-Dependent Plasticity: A Hebbian Rule. *Annu. Rev. Neurosci.*.

Chechik, G. (2003). Spike-timing-dependent plasticity and relevant mutual information maximization. *Neural Compuatation*, *15*(7), 1481-1510.

Clopath, C., Ziegler, L., Vasilaki, E., Buesing, L., & Gerstner, W. (2008). Tag-trigger-consolidation: A model of early and late long-term-potentiation and depression. *PLoS Comput Biol*, *4*(12).

Cooper, L., Intrator, N., Blais, B., & Shouval, H. Z. (2004). *Theory of cortical plasticity.* Singapore: World Scientific.

Cover, T. M., & Thomas, J. A. (1991). *Elements of information theory.* New York: Wiley.

Creutzig, F., & Sprekeler, H. (2008). Predictive coding and the slowness principle: An information-theoretic approach. *Neural Computation*, *20*(4), 1026-1041. (doi:10.1162/neco.2008.01-07-455)

DeFelipe, J., & Fariñas, I. (1992). The pyramidal neuron of the cerebral cortex: morphological and chemical characteristics of the synaptic inputs. *Prog Neurobiol.*, *39*(6), 563-607.

Denk, W., & Horstmann, H. (2004). Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure. *PLoS Biol*, *2(11)*, e329. doi:10.1371/journal.pbio.0020329.

Derrida, B., & Pomeau, Y. (1986). Random networks of automata: A simple annealed approximation. *Europhysics Letters*, *1*(2), 45-49.

Derrida, B., & Stauffer, D. (1986). Phase Transitions in Two-Dimensional Kauffman Cellular Automata. *Europhys Lett*, *2*, 739-745.

Destexhe, A., Rudolph, M., & Pare, D. (2003). The high-conductance state of neocortical neurons in vivo. *Nat. Rev. Neurosci.*, *4*(9), 739–751.

Dudek, S. M., & Bear, M. F. (1993). Bidirectional long-term modification of synaptic effectiveness in the adult and immature hippocampus. *J. Neuroscience*, *13*, 2910-2918.

Fregnac, Y., & Shulz, D. (1999). Activity-dependent regulation of receptive field properties of cat area 17 by supervised hebbian learning. *J. Neurobiol*, *41*, 69-82.

Frey, U., & Morris, R. (1997). Synaptic tagging and long-term potentiation. *Nature*, *385*, 533 - 536.

Friedman, N., Mosenzon, O., Slonim, N., & Tishby, N. (2001). Multivariate information bottleneck. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence* (pp. 152–161). Menlo Park, USA: AAAI Press.

Froemke, R. C., Merzenich, M. M., & Schreiner, C. E. (2007). A synaptic memory trace for cortical receptive field plasticity. *Nature*, *450*, 425-429.

Ganguli, S., Huh, D., & Sompolinsky, H. (2008). Memory traces in dynamical systems. *PNAS*, *15*(48), 18970-18975.

Gerstner, W., Kempter, R., van Hemmen, L., & Wagner, H. (1996). A neuronal learning rule for sub-millisecond temporal coding. *Nature*, *383*, 76-78.

Gerstner, W., & Kistler, W. M. (2002). *Spiking neuron models*. Cambridge: Cambridge University Press.

Gütig, R., Aharonov, R., Rotter, S., & Sompolinsky, H. (2003). Learning input correlations through non-linear temporally asymmetric Hebbian plasticity. *Journal of Neurosci.*, *23*, 3697–3714.

Guyonneau, R., VanRullen, R., & Thorpe, S. (2005). Neurons tune to the earliest spikes through stdp. *Neural Computation*, *17*(4), 859–879.

Haeusler, S., & Maass, W. (2007). A statistical analysis of information processing properties of lamina-specific cortical microcircuit models. *Cerebral Cortex*, *17*(1), 149-162.

Hardie, J., & Spruston, N. (2009). Synaptic depolarization is more effective than back-propagating action potentials during induction of associative long-term potentiation in hippocampal pyramidal neurons. *J. Neurosci*, *2009*, 29: 3233 - 3241.

Harremoes, P., & Tishby, N. (2007, June). The information bottleneck revisited or how to choose a good distortion measure. In *Proceedings of the IEEE International Symposium on Information Theory, 2007 (ISIT 2007)* (p. 566-570). : IEEE.

Hebb, D. O. (1949). *The Organization of Behavior*. New York: Wiley.

Hee, S. G., Ziburkus, J., Huang, S., Song, L., Kim, I. T., Takamiya, K., et al. (2007). Neuromodulators Control the Polarity of Spike-Timing-Dependent Synaptic Plasticity. *Neuron*, *55*, 919-929.

Hinton, G. (2007). To recognize shapes, first learn to generate images. *Progress in brain research*, *165*, 535.

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

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. USA*, *79*, 2554–2558.

Hyvaerinen, A., Karhunen, J., & Oja, E. (2001). *Independent component analysis*. Wiley-Interscience.

Ince, R. A. A., Petersen, R. S., Swan, D. C., & Panzeri, S. (2009). Python for information theoretic analysis of neural data. *Frontiers in Neuroinformatics*(4).

Intrator, N., & Cooper, L. N. (1992). Objective function formulation of the BCM theory of visual cortical plasticity: statistical connections, stability conditions. *Neural Networks*, *5*, 3-17. Available from `citeseer.ist.psu.edu/intrator92objective.html`

Izhikevich, E. M. (2007). Solving the Distal Reward Problem through Linkage of STDP and Dopamine Signaling. *Cerebral Cortex*, bhl152. Available from `http://cercor.oxfordjournals.org/cgi/content/abstract/bhl152v1`

Izhikevich, E. M., & Edelman, G. M. (2008). Large-scale model of mammalian thalamocortical systems. *Proceedings of the National Academy of Sciences*, *105*, 3593-3598. Available from `http://www.pnas.org/cgi/content/abstract/0712231105v1`

Jadhav, S. P., Wolfe, J., & Feldman, D. E. (2009). Sparse temporal coding of ele-

mentary tactile features during active whisker sensation. *Nature Neuroscience*, doi:10.1038/nn.2328.

Jaeger, H. (2001). *The "echo state" approach to analyzing and training recurrent neural networks* (GMD Report No. 148). : German National Research Center for Information Technology.

Jaeger, H. (2002). *Short term memory in echo state networks* (GMD Report No. 152). German National Research Center for Information Technology.

Jaeger, H. (2007). Echo state networks. *Scholarpedia*, *2*(9), 2330.

Jaeger, H., & Haas, H. (2004). Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science*, *304*, 78–80.

Jaeger, H., Lukosecicius, M., Popovici, D., & Siewert, U. (2007). Optimization and applications of echo state networks with leaky-integrator neurons. *Neural Networks*, *20*(3), 335-352.

Joshi, P., & Maass, W. (2005). Movement generation with circuits of spiking neurons. *Neural Computation*, *17*(8), 1715–1738.

Joshi, P., & Triesch, J. (2008). A globally asymptotically stable plasticity rule for firing rate homeostasis. In *Proceeding of the International Conference on Neural Networks (ICANN).*

Katok, A., & Hasselblatt, B. (1995). *Introduction to the Modern Theory of Dynamical Systems*. Cambridge Univ. Press.

Kauffman, S. A. (1969). Metabolic stability and epigenesis in randomly connected nets. *J. Theoret. Biol.*, *22*, 437.

Klampfl, S., Legenstein, R., & Maass, W. (2007). Information bottleneck optimization and independent component extraction with spiking neurons. In *Proc. of NIPS 2006, Advances in Neural Information Processing Systems* (Vol. 19, pp. 713–720). : MIT Press.

Klampfl, S., Legenstein, R., & Maass, W. (2009). Spiking neurons can learn to solve information bottleneck problems and to extract independent components. *Neural Computation*, *21*(4), 911–959.

Kozloski, J., & Cecchi, G. A. (2008). Topological effects of spike timing-dependent plasticity. *arxiv.org*, *abs*, 0810.0029.

Langton, C. G. (1990). Computation at the edge of chaos. *Physica D*, *42*, 12–37.

Laughlin, S., de Ruyter van Steveninck, R., & Anderson, J. (1998). The metabolic cost of neural information. *Nature Neuroscience*, *1*, 36-41.

Lazar, A., Pippa, G., & Triesch, J. (2007). Fading memory and time series prediction in recurrent networks with different forms of plasticity. *Neural Networks*, *20*, 312-322.

Lefort, S., Tomm, C., Sarria, J., & Petersen, C. (2009). The excitatory neuronal network of the c2 barrel column in mouse primary somatosensory cortex. *Neuron*, *61*, 301-316.

Legenstein, R., & Maass, W. (2007a). Edge of chaos and prediction of computational performance for neural microcircuit models. *Neural Networks*, *20*(3), 323–334.

Legenstein, R., & Maass, W. (2007b). What makes a dynamical system computationally powerful? In S. Haykin, J. C. Principe, T. Sejnowski, & J. McWhirter (Eds.), *New*

Directions in Statistical Signal Processing: From Systems to Brains (pp. 127–154). MIT Press.

Legenstein, R., Markram, H., & Maass, W. (2003). Input prediction and autonomous movement analysis in recurrent circuits of spiking neurons. *Reviews in the Neurosciences (Special Issue on Neuroinformatics of Neural and Artificial Computation)*, *14*(1–2), 5–19.

Legenstein, R., Naeger, C., & Maass, W. (2005). What can a neuron learn with spike-timing dependent plasticity. *Neural Computation*, *17*, 2337-2382.

Lennie, P. (2003). The cost of cortical computation. *Current Biology*, *13*, 493–497.

Levy, N., Horn, D., Meilijson, I., & Ruppin, E. (2001). Distributed synchrony in a cell assembly of spiking neurons. *Neural Networks*, *14*(6-7), 815 - 824.

Levy, W. B., & Baxter, R. A. (1996). Energy efficient neural codes. *Neural Computation*, *8*(3), 531-543.

Linsker, R. (1989). How to generate ordered maps by maximizing the mutual information between input and output signals. *Neural Computation*, *1*(3), 402-411.

Lisman, J., & Spruston, N. (2005). Postsynaptic depolarization requirements for LTP and LTD: a critique of spike timing-dependent plasticity. *Nature Neuroscience*, *8*(7), 839–841.

Lisman, J., & Zhabotinsky, A. (2001). A model of synaptic memory: A CaMKII/PP1 switch that potentiates transmission by organizing an AMPA receptor anchoring assembly. *Neuron*, *31*, 191-201.

Lubenov, E. V., & Siapas, A. G. (2008). Decoupling through synchrony in neuronal circuits with propagation delays. *Neuron*, *58*, 118-131.

Lukosecicius, M., & Jaeger, H. (2007). *Overview of reservoir recipes* (Tech. Rep. No. 11). Jacobs University Bremen.

Luque, B., & Solé, R. V. (2000). Lyapunov exponents in random Boolean networks. *Physica A*, *284*, 33-45.

Maass, W., Natschlaeger, T., & Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, *14*(11), 2531-2560.

Maass, W., & Sontag, E. D. (2000). Neural systems as nonlinear filters. *Neural Computation*, *12*(8), 1743–1772.

Malenka, R. C., & Bear, M. F. (2004). LTP and LTD: An embarassment of riches. *Neuron*, *44*, 5–21.

Markram, H., Lübke, J., Frotscher, M., & Sakmann, B. (1997). Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs. *Science*, *275*, 213–215.

Mayor, J., & Gerstner, W. (2005). Signal buffering in random networks of spiking neurons: Microscopic versus macroscopic phenomena. *Physical Review E*, *72*(051906).

Meffin, H., Besson, J., Burkitt, A. N., & Grayden, D. B. (2006). Learning the structure of correlated synaptic subgroups using stable and competitive spike-timing-dependent plasticity. *Physical Review E*, *73*.

Miller, K. D. (1994). A model for the development of simple cell receptive fields and the ordered arrangement of orientation columns through activity dependent com-

petition between ON- and OFF-center inputs. *J. Neurosci.*, *14*, 409-441.

Mitchell, M., Hraber, P. T., & Crutchfield, J. P. (1993). Revisiting the edge of chaos: Evolving cellular automata to perform computations. *Complex Systems*, *7*, 89–130.

Morrison, A., Aertsen, A., & Diesmann, M. (2007). Spike-timing dependent plasticity in balanced random networks. *Neural Computation*, *19*, 1437-1467.

Natschlaeger, T., Bertschinger, N., & Legenstein, R. (2005). At the edge of chaos: Real-time computations and self-organized criticality in recurrent neural networks. In L. K. Saul, Y. Weiss, & L. Bottou (Eds.), *Advances in Neural Information Processing Systems (NIPS) 17* (p. 145-152). Cambridge, MA: MIT Press.

Nevian, T., & Sakmann, B. (2006). Spine ca2+ signaling in spike-timing-dependent plasticity. *J. Neurosci.*, *26*(43), 11001-11013. Available from `http://www.jneurosci.org/cgi/content/abstract/26/43/11001`

Ngezahayo, A., Schachner, M., & Artola, A. (2000). Synaptic activity modulates the induction of bidirectional synaptic changes in adult mouse hippocampus. *The Journal of Neuroscience*, *20(7)*, 2451-2458.

O'Connor, D., Wittenberg, G., & Wang, S. (2005). Dissection of Bidirectional Synaptic Plasticity Into Saturable Unidirectional Processes. *Journal of Neurophysiology*, *94*, 1565–1573.

Oja, E. (1982). A simplified neuron model as a principal component analyzer. *J. Mathematical Biology*, *15*, 267-273.

Olshausen, B. A., & Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, *381*, 607–609.

Packard, N. (1988). Adaption towards the edge of chaos. In J. A. S. Kelso, A. J. Mandell, & M. F. Shlesinger (Eds.), *Dynamic patterns in complex systems* (pp. 293–301). World Scientific.

Papoulis, A. (1991). *Probability, Random Variables and Stochastic Processes* (3rd ed.). : McGraw-Hill.

Parra, L., Beck, J., & Bell, A. (2009). On the maximization of information flow between spiking neurons. *Neural Computation*, 1-19.

Pfister, J.-P., & Gerstner, W. (2006). Triplets of spikes in a model of spike timing-dependent plasticity. *J. Neuroscience*, *26*, 9673-9682.

Recanzone, G. H., Schreiner, C. E., & Merzenich, M. M. (1993). Plasticity in the frequency representation of primary auditory cortex following discrimination training in adult owl monkeys. *The Journal of Neuroscience*, *13*, 87-103.

Risken, H. (1996). *The Fokker-Planck Equation* (3rd ed.). : Springer.

Roberts, P., & Bell, C. (2000). Computational consequences of temporally asymmetric learning rules: II. Sensory image cancellation. *Computational Neuroscience*, *9*, 67-83.

Rumelhart, D., Hinton, G., & Williams, R. (1986). Learning representations by back-propagating errors. *Nature*, *323*, 533–536.

Saudargiene, A., Porr, B., & Wörgötter, F. (2003). How the shape of pre- and postsynaptic signals can influence stdp: A biophysical model. *Neural Computation*, *16*, 595-626.

Schrauwen, B., Wardermann, M., Verstraeten, D., Steil, J. J., & Stroobandt, D. (2008, 1). Improving reservoirs using intrinsic plasticity. *Neurocomputing*, 1159-1171.

Sejnowski, T. J., & Tesauro, G. (1989). The hebb rule for synaptic plasticity: algorithms and implementations. In J. H. Byrne & W. O. Berry (Eds.), *Neural Models of Plasticity* (p. 94-103). : Academic Press.

Senn, W., Tsodyks, M., & Markram, H. (2001). An algorithm for modifying neurotransmitter release probability based on pre- and postsynaptic spike timing. *Neural Computation*, *13*, 35-67.

Shamir, O., Sabato, S., & Tishby, N. (2008). Learning and generalization with the information bottleneck. In *International Symposium on AI and Mathematics (ISAIM)-2008.* : .

Shmulevich, I., Dougherty, E., Kim, S., & Zhang, W. (2002). Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks. *Bioinformatics*, *18*(2), 261-274.

Shouval, H. Z., Bear, M. F., & Cooper, L. N. (2002). A unified model of nmda receptor dependent bidirectional synaptic plasticity. *Proc. Natl. Acad. Sci. USA*, *99*, 10831-10836.

Singh, C. K., Prasad, S. H., & Balsara, P. T. (2007). VLSI Architecture for Matrix Inversion using Modified Gram-Schmidt based QR Decomposition. In *VLSID '07: Proceedings of the 20th International Conference on VLSI Design held jointly with 6th International Conference* (pp. 836–841). Washington, DC, USA: IEEE Computer Society.

Sjöström, P., Turrigiano, G., & Nelson, S. (2003). Neocortical ltd via coincident activation of presynaptic nmda and cannabinoid receptors. *Neuron*, *39*, 641-654.

Sjöström, P. J., & Häusser, M. (2006). A Cooperative Switch Determines the Sign of Synaptic Plasticity in Distal Dendrites of Neocortical Pyramidal Neurons. *Neuron*, 227-238.

Sjöström, P. J., Rancz, E. A., Roth, A., & Häusser, M. (2008). Dendritic Excitability and Synpatic Plasticity. *Physiol. Rev.*.

Sjöström, P. J., Turrigiano, G. G., & Nelson, S. B. (2001). Rate, timing and cooperativity jointly deterimine cortical synaptic plasticity. *Neuron*, *32*, 1149-1164.

Slonim, N., & Tishby, N. (1999). Agglomerative information bottleneck. In *Advances in Neural Information Processing Systems (NIPS).* Cambridge (MA): MIT Press.

Slonim, N., & Tishby, N. (2001). The power of word clustering for text classification. In *Proceedings of the European Colloquium on IR Research, ECIR 2001.* : .

Slonim, N., & Weiss, Y. (2003). Maximum likelihood and the information bottleneck. *Advances In Neural Information Processing Systems (NIPS)*, 351–358.

Solé, R., Luque, B., & Kauffman, S. (2000). *Phase transitions in random networks with multiple states* (Tech. Rep. No. 00-02-011). Santa Fe Institute.

Song, S., & Abbott, L. F. (2001). Cortical development and remapping through spike timing-dependent plasticity. *Neuron*, *32*, 339-350.

Song, S., Sjöström, P., Reigl, M., Nelson, S., & Chklovskii, D. (2005). Highly nonrandom features of synaptic connectivity in local cortical circuits. *PLoS Biology*, *3*, 507-519.

Stuart, G. J., & Häusser, M. (2001). Dendritic coincidence detection pf EPSPs and action potentials. *Nature*, *4*(1).

Tishby, N., Pereira, F. C., & Bialek, W. (1999). The information bottleneck method. In *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing* (pp. 368–377).

Toyoizumi, T., Pfister, J.-P., Aihara, K., & Gerstner, W. (2005). Generalized Bienenstock-Cooper-Munro rule for spiking neurons that maximizes information transmission. *Proc. Natl. Acad. Sci. USA*, *102*, 5239–5244.

Toyoizumi, T., Pfister, J.-P., Aihara, K., & Gerstner, W. (2007). Optimality Model of Unsupervised Spike-Timing Dependent Plasticity: Synaptic Memory and Weight Distribution. *Neural Computation*, *19*(3), 639–671.

Trachtenberg, J. T., Chen, B. E., Knott, G. W., Feng, G., Sanes, J. R., Welker, E., et al. (2002). Long-term in vivo imaging of experience-dependent synaptic plasticity in adult cortex. *Nature*, *420*, 788-794.

Triesch, J. (2007). Synergies between intrinsic and synaptic plasticity mechanisms. *Neural Compuatation*, *19*(4), 885-909.

Tsodyks, M., & Markram, H. (1997). The neural code between neocortical pyramidal neurons depends on neurotransmitter release probability. *Proc. Nat. Acad. Sci. USA*, *94*, 719-23.

Turrigiano, G., & Nelson, S. (2004). Homeostatic plasticity in the developing nervous system. *Nature Reviews Neuroscience*, *5*, 97-107.

Vandoorne, K., Dierckx, W., Schrauwen, B., Verstraeten, D., Baets, R., Bienstman, P., et al. (2008). Toward optical signal processing using photonic reservoir computing. *Optics Express*, *16*(15), 11182–11192.

Vapnik, V. N. (1998). *Statistical Learning Theory.* John Wiley (New York).

Verstraeten, D., Schrauwen, B., D'Haene, M., & Stroobandt, D. (2007). A unifying comparison of Reservoir Computing methods. *Neural Networks*, *20*, 391-403.

Verstraeten, D., Schrauwen, B., Stroobandt, D., & Campenhout, J. V. (2005). Isolated word recognition with the liquid state machine: a case study. *Information Processing Letters*, *95*(6), 521–528.

Verstraeten, D., Souza, S. Xavier-de, Schrauwen, B., Suykens, J., Stroobandt, D., & Vandewalle, J. (2008, 9). Pattern classification with CNNs as reservoirs. In *Proceedings of the International Symposium on Nonlinear Theory and its Applications (NOLTA).*

Vogels, T. P., Rajan, K., & Abbott, L. (2005). Neural Networks Dynamics. *Annual Review of Neuroscience*, *28*, 357-376.

Wang, H., Gerkin, R., Nauen, D., & Bi, G. (2005). Coactivation and timing-dependent integration of synaptic potentiation and depression. *Nat. Neurosci.*, *8*(2), 187-93.

Weiss, R. (2007). *Predictive Information as a Criterion to Linear Dynamical Systems Reduction.* Unpublished master's thesis, The Racah Institute of Physics, The Hebrew University, Jerusalem, Israel.

White, O. L., Lee, D. D., & Sompolinsky, H. (2004). Short-term memory in orthogonal neural networks. *Phys. Rev. Letters*, *92*(14), 148102.

Wiskott, L., & Sejnowski, T. (2002). Slow feature analysis: Unsupervised learning of

invariances. *Neural Computation*, *14*(4), 715–770.

Wolfram, S. (1984). Universality and complexity in cellular automata. *Physica D*, *10*, 1–35.

Yuste, R., & Bonhoeffer, T. (2004). Genesis of dendritic spines: insights from ultrastructural and imaging studies. *Nat Rev Neurosci*, *5*(1), 24–34.

Zinn-Justin, J. (2003). *Phase Transitions and Renormalization Group: from Theory to Numbers*. Oxford: Oxford Univ. Press.