Mathematics of the Mind: Probabilistic Inference and Learning in Spiking Neuronal Circuits

Bernhard Nessler

Institut für Grundlagen der Informationsverarbeitung

Technische Universität Graz

A thesis submitted for the degree of

PhilosophiæDoctor (PhD) in Telematics

February 2014

1. Reviewer: o.Univ.-Prof. Dipl.-Ing. Dr.rer.nat. Wolfgang Maass

2. Reviewer: o.Univ.-Prof. Dipl.-Ing. Dr.techn. Peter Auer

Day of the defense: 21 February 2014

Signature from head of PhD committee:
Assoc.Prof. Dipl.-Ing. Dr.techn. Denis Helic

To Klaudia, Felix, Stephanie, and Johanna

Acknowledgements

I would like to acknowledge first my colleagues and collaborators; Michael Pfeiffer for helping me to get started working in a scientific way and writing papers, Lars Büsing for his patience in uncountable mathematical discussions, Johannes Bill and Stefan Habenschuss for the great collaboration and exchange of ideas and for the friendship and the personal sustain they both gave me also in difficult times.

I would like to thank my wife Klaudia and my family for keeping at it and loving me despite the countless evenings, nights and weekends that I devoted into this work. And I would like to thank my wife explicitly for taking care of our children in these times.

Abstract

There is strong evidence for two crucial facts that seem to revolutionize the traditional school of computational neuroscience and their models of cortical processes. Firstly, we see in experiments that the repeated application of the same sensory stimulus produces varying responses in the measured spike patterns of the respective neurons in the cortex. This stochastic trial-to-trial variability seems to be inherent to the neuronal processes. Secondly, we see that humans and more general mammals exhibit near statistically optimal behavior in their decisions and estimations about the world. This optimality in the sense of Bayesian probabilistic inference means that our brain correctly takes into account the omnipresence of uncertainty in our experience of the world due to limitations of our sensing organs and correctly merges these uncertain evidences with priorly learned knowledge of the world.

The classic school of connectionism models the brain as an artificial neural network that consists in single deterministic neurons, resulting in a complex system of differential equations. Yet this traditional model fails both in explaining the variability of the neuronal processes and in the possibility to exhibit Bayes optimal learning and decision-making.

In this thesis I present a collection of results that aims to bridge that gap. The starting point is a supervised or reinforcement learning approach to learning Bayesian optimal inference in feed-forward artificial networks. This approach is developed further to a stochastically spiking Winner-Take-All network that learns to adapt its inference process in a Bayesian optimal way using only fully local STDP. This approach is then generalized to the neural sampling theory as a new paradigm for modeling the cognition process. It is shown that it is the inherent processing variability of neuronal networks that enables them to carry out Bayesian probabilistic inference using a stochastic Markov-Chain Monte-Carlo sampling algorithm. Thus it turns out that the two "problems" in fact mutually provide their solutions.

Zusammenfassung

Es gibt starke Hinweise für zwei wesentliche Tatsachen, die die traditionelle Schule der Theoretischen Neurowissenschaften und deren kortikale Prozessmodelle zu revolutionieren scheint. Erstens sehen wir in Experimenten, dass die wiederholte Anwendung ein und desselben sensorischen Stimulus unterschiedliche Spike-Mustern in den jeweils zuständigen Neuronen im Cortex hervorruft. Diese Trial-to-Trial Variabilität scheint eine grundlegende Eigenschaft der neuronalen Prozesse zu sein. Zweitens beobachten wir, dass Menschen und ganz allgemein Säugetiere ein statistisch nahezu optimales Verhalten in ihren Entscheidungen und Erwartungen über ihre Umwelt zeigen. Das heißt, dass unser Gehirn die inhärente Ungenauigkeit unserer beschränkten Sinnesorgane berücksichtigt und diese unsicheren Wahrnehmungen in Bayes'sch optimaler Weise mit früher gelerntem Wissen über die Umwelt verbindet.

Die klassische Schule des Konnektionismus stellt das Gehirn als künstliches neuronales Netz dar, das aus einzelnen deterministischen Neuronen besteht. Dies führt zu einem komplexen System von Differentialgleichungen. Dieses traditionelle Modell versagt jedoch sowohl darin, die Variabilität der neuronalen Prozesse zu erklären, als auch darin, Bayes'sch optimales Lernen und Entscheiden hervorzubringen.

Diese Doktorarbeit besteht aus einer Reihe von Ergebnissen, die darauf abzielen diese Lücke zu schließen. Den Ausgangspunkt bildet ein Ansatz von überwachtem oder verstärkendem Lernen in sequentiellen neuronalen Netzwerken. Dieser Ansatz wird sodann zu einem stochastischen, spikenden Winner-Take-All Netzwerk weiterentwickelt, das mit rein lokalem STDP seinen Inferenzprozess optimal einzustellen lernt. Sodann wird dieser Ansatz zur Theorie des neuronalen Samplings verallgemeinert, einem neuen Paradigma zur Modellierung des Denkprozesses. Gerade die inhärente Variability erlaubt dem Netzwerk Bayes'sche Inferenz mithilfe eines MCMC Samplingalgorithmus auszuführen. So stellt sich heraus, dass die beiden "Probleme" tatsächlich wechselseitig ihre Lösungen darstellen.

Contents

List of Figures

List of Tables

LIST OF TABLES

1

Introduction

The Brain

One of the most fascinating complex systems that we know is the human brain, and that is true for two reasons. The first reason is simply its incredible degree of complexity, a feature that it shares with a lot of other systems or fields like the quantum physics, the astrophysics, the weather and climate dynamics or just other kinds of brains, like the brain of a frog or a cat. The second reason is a very specific one. It is the fact that our own thoughts on the way of analyzing our brain necessarily have to be build up and emerge within that very brain. Even if we use tools like paper and pencil, or even computers to help our thinking process, our goal is still to come to a real understanding, which means to realize kind of an abstract model of our brain within itself. This is what distinguishes investigating the functionality of our brain from analyzing or modeling all other kind of complex systems. On the philosophical basis of a strong anthropocentric approach one might argue that a similar reciprocity would in principle also apply to the study of the human genome or the quantum physics, but it does so in much more indirect way.

The brain acquires a model of the environment and of its own body within this environment. The working brain is not only able to solve immediate problems – like grasping and eating food or chasing the prey in sight – but also to develop plans like they are necessary for collectively organized hunts or the cultivation of land and animals. It is clear that these skills - especially the more demanding ones do not emerge from a single brain by itself. It is only by observing their parents that animals learn such complicated things like collectively hunting or using tools for cracking nuts. The most

elaborated skill of our human brain in the sense of the highest level of abstraction is the ability to use speech for communications. This enables us to transfer complex thoughts and ideas from one individual to another. It is on the basis of this repertoire of verbal abstractions that we were able to develop our high cognitive achievements.

Bottom Up

The most fundamental question in neuroscience and cognitive science is the quest for the underlying principles that enable the brain to acquire such demanding capabilities. In a bottom up analyzing paradigm experimental neuroscience has in the last decades reported an overwhelming number of detailed experiments about single neurons and synaptic connections. As a consequence there is a fairly good understanding of the function of single neurons and synapses at the level of the course of the ionic currents and the membrane potential in response to the presynaptic spikes and the resulting generation of output spikes [38]. This understanding is formulated in a number of models at different levels of abstraction, ranging from the Hodgkin-Huxley model, which is able to reflect the detailed anatomic structure of the neuron and its dendritic and axonal tree, to the LIF-models that abstract from the anatomic structure but still model realistic membrane potentials and currents, to spike response models that functionally emerge from a simple linearization of the former models by abstracting from the precise electrical description, resulting in a very simple description of the behavior of the neurons at the level of input and output spikes.

The dynamics of the chemical synapses between neurons seems to be much more difficult to describe in meaningful levels of abstraction. This is true mainly because the process of learning on large time scales is contributed to lasting changes in the synaptic connections whereas the electrical dynamics of the individual neurons is thought to be more or less fixed, according to the current view in neuroscience. There exist elaborated models about short term synaptic plasticity, facilitation and depression, from an understanding at the chemical level up to an abstract dynamic description, but these models assume that the strength of the synapses return to some individual fixed values after some time. First theories about the long term plasticity of synapses date back 1949, when Donald Hebb postulated that a synapse is strengthened "if the firing activity of the presynaptic neuron repeatedly or persistently takes part in firing the postsynaptic neuron" [96]. The current formal version of this level of abstraction is the

model of spike-timing dependent plasticity (STDP), which postulates a potentiation if the presynaptic spike is before and a depression if the presynaptic spike comes after the postsynaptic spike, with the intensity of both mechanisms fading out exponentially with the absolute timing difference between the two spikes [1]. More precise models take into account the dependency of the timing of a third spike, as in the triplet rule [175], or the dependency of further neurotransmitters that may modulate the effect of plasticity. A closer look at the chemical level reveals that the changes in the synapse do not happen instantaneously but merely are just prepared or flagged in response to the instantaneous pre- and postsynaptic activity, such that these intended changes may or may not be consolidated at a later time which has important consequences for possible functional models [109]. The puzzle is further complicated by the fact that the connection between two neurons most often does not consist in a single synapse but in a number of parallel synapses, possibly at different branches of the dendritic and the axonal tree, respectively. It is questionable if, as in the STDP model, all these details can be abstracted from in the aim to understand the implementation of the function of the whole system. However, most current higher level models do so or only take into account some of the details.

On and above the level of local circuits the experimental results from classical neuroscience become extremely difficult to interpret. Great advances have been made in order to localize functions and functional maps in the brain, i.e. the position of neurons that respond most directly and most specifically to certain stimuli. Such experiments revealed e.g. fixed maps of stimulus-selective receptive fields in V1 [105] and re-mappable place cells in hippocampus that adapt very quickly to new environments. There are also numerous studies about the anatomical connections between neurons, converging in the aim to discover the full connectome of the human brain. Despite the huge number of detail information that is gained by these experiments, they revealed only very little about the functional principles of the brain in the sense of an algorithmic description of the information processing. One of the few functional abstractions at the level of cortical microcircuits is the hypothesis of so called Winner-Take-All (WTA) structures [52], which are proposed to be an ubiquitous functional principle in the cerebral cortex. A very generalized form of these WTA structures in one pillar of the theories that are developed in the present work.

One reason for the lack of experimental evidences for functional abstractions at this intermediate level of cortical microcircuits is the fact that classical experimental investigation methods to measure electrical signals are very limited in the number of neurons that can be recorded from simultaneously. This reaches from a few single neurons - two or three - that can be patch-clamped in order to record precise membrane potentials to some tens of neurons from which single spikes can be recorded using multi-electrodes and spike-sorting. Newest techniques using two-photon microscopy and calcium imaging can in principle be used to visualize the activity of hundreds of neurons at the same time but doing so reduces the time resolution of the acquisition. Of course there are also other experimental techniques, like e.g. measuring the local field potential or applying voltage sensitive dyes or functional magnet-spin resonance imaging. These techniques cannot resolve to the level of detail of individual neurons, and though they are very helpful in describing *what* large scale activities are going on but give little insight into the detailed neuronal mechanisms *why* this is going on.

The small numbers of neurons in current experimental settings have to be seen in relation to the billions of neurons in the mammalian brain - around 85 billions in humans of which 16 billions are located in the cerebral cortex [12]. This small scope of our experimental view explains the difficulty in discovering functional principles of this huge network.

Top Down

If we call the experimental neuroscience approach a bottom up way of analysis, then the corresponding top-down approach can be found in cognitive science, which tries to find the principles of cognition and information processing in the brain at it's highest level mainly through behavioral experiments. Traditionally cognitive science tried to model cognitive processes in terms of associative memories, abstract symbolic processes, dynamical systems or through connectionist artificial neural networks (ANNs). All these essentially deterministic models fail to explain the variability that is observed in neuronal processes. This variability can sometimes be observed at a behavioral level and more essentially is observed on the level of the neuronal responses in repeated trials with identical stimuli. It is well known that the generation of spikes in dependence of the membrane potential is an almost deterministic process, i.e. the value of the spiking threshold is a very precise one and thus the spike generation itself is probably not the

cause of the observed neuronal variability. Even though other causes are thinkable it seems most plausible that it stems from the inherent stochastic nature of the remaining elements of the neuronal processing system, more precisely from the stochasticity in the synaptic vesicle release and in the dendritic ion channels.

Over these last two decades human cognition was increasingly modeled with respect to this inherent stochasticity by means of Bayesian probabilistic models. The usage of Bayesian probabilistic models for cognition provides a big advantage over deterministic models like traditional connectionist ANNs or abstract symbolic models. The probabilistic models are naturally able to capture and model noise in sensory inputs as input uncertainty and explain the variability in the processing as the result of a prior distribution that is shaped from hitherto experience. It was shown in a wide variety of tasks that the human inference capabilities can be reasonably modeled such that the observable behavior is near optimal in a Bayesian sense. The tasks for which this is experimentally confirmed range from visual perception [122, 123] to decision making [128, 212] to sensorimotor control [129], just to name a few of them. In [232] the authors measured spiking activity in the area LIP during a complex decision process. They showed that the measured spiking activity of one such decision neuron correspond to the log-likelihood of the Bayesian model. [87] shows that the brain acquires internal models of prior distributions about properties of specific things and is able to use these priors in order to carry out formerly unknown inference tasks with close to Bayesian-optimal performance. Further experiments were able to underpin the assumption that also the learning itself in the brain exhibits the fundamental characteristics of Bayesian optimality [169].

The most recent theory in the quest of neural mechanisms or algorithms that are likely to explain this Bayesian optimality is the so-called sampling hypothesis [104]. Instead of postulating specific complex neuronal circuits that implement deterministic inference algorithms (like [183]) the sampling hypothesis assumes that the sequence of states of the cortex' network form a sample-based representation of the posterior distribution according to a Markov-Chain Monte-Carlo process. The theory predicts that the variability in the neuronal activity is specifically used in order to enable the neuronal network to represent probability distributions by sampling. The advantage of this theory over concurrent hypotheses for the implementation of Bayesian inference (like Probabilistic population codes [143]) is the fact that it is able to explain why it is

not necessary to represent the entire probability distribution at a time in the neuronal code.

My contribution

The present thesis, i.e. the published papers that jointly constitute this thesis aim to contribute to this field and find models that bridge the gap between neuronal models on one side and the high-level abstract model of cognition as Bayesian inference and learning on the other side.

In the first work, in chapter 2, which is a joint work with Michael Pfeiffer and Wolfgang Maass [158], we present a neuronal plausible model of how neurons could learn Bayesian optimal decision making. The heart of the model is a synaptic learning rule that – in accordance with the Hebbian postulate – increases its weight if both the pre and the postsynaptic neuron are active together, and decreases its weight otherwise. The key feature of the learning rule is the dependency of the amount of weight increase on the current weight value in a negative exponential way, i.e. the increase is small if the weight is already strong and more significant if the weight is weak. We show that a simple feed-forward unit implements the Naive Bayes classifier if the inputs are encoded by a population code. More than that we show that a simple fixed preprocessing circuit enables the learning of Bayesian optimal decisions also in the general case of arbitrary statistical dependencies and we extend this theory to include a reward modulation signal in the learning rule. Finally we use that model in order to explain a seminal experimental result of Yang and Shadlen [232] showing that the measured activity of neurons in the LIP of monkeys are proportional to the log-likelihood of the respective decision.

The derivation of the learning rule, the convergence proofs, the learning rate adaptation and preliminary simulation experiments were carried out by me alone, initially based on a preliminary idea of Wolfgang Maass. The final experiments as well as the adaptation of the learning rule for reward-modulated learning was made by Michael Pfeiffer in close collaboration with me. The paper was written by all three author together.

The second paper, in chapter 3 is a joint work with Michael Pfeiffer, Rodney Douglas, and Wolfgang Maass [173]. It deepens the understanding of the reward-modulated learning of Bayesian optimal decisions using the same local synaptic learning paradigm

that was already introduced shortly in the first paper. We discuss different action selection strategies and show how they are implemented by a Winner-Take-All (WTA) circuit. We analyze the convergence properties of the learning rule and of a linear variant. We then explore the extended population codes and a generalization to continuous variables. In simulation experiments we compare the learning speed of our rule with the non-local learning Rescorla-Wagner rule and clearly see the advantage of our rule both in convergence speed and in the final performance. Again we discuss the experiment of Yang and Shadlen and provide extensive simulation results that show striking similarities to the measurements taken from the LIP of the monkeys.

The derivation of the learning rule, the convergence proofs, the learning rate schedules and the learning rate adaptation algorithm were developed by me alone, based on a preliminary idea of Wolfgang Maass. The reward modulation of the learning rule was developed by Michael Pfeiffer in close collaboration with me. The extensive experimental work was carried out by Michael Pfeiffer alone. A first version of the text of the paper was written by Michael Pfeiffer, Wolfgang Maass, and me. This was heavily reworked by Rodney Douglas and Michael Pfeiffer during the revisions.

The third paper, in chapter 4 appeared in two steps. A first extended abstract was presented at NIPS '09 [159], the final complete work was published 2012 in PLoS Computational Biology. This is joint work with Michael Pfeiffer, Lars Büsing, and Wolfgang Maass. We achieved to draw an important connection in this work between realistic spiking neuronal circuits and Bayesian inference and learning. Inspired by the previous work on supervised or reward-modulated learning we derive an unsupervised model-based learning mechanism that is readily implemented in a spiking WTA-structure. We prove that a variant of the currently widely accepted model of learning in biological neurons, spike-timing dependent plasticity (STDP) has the effect of carrying out Expectation Maximization (EM) learning in a simple Bayesian Network model. We coin the important concept, that a single spike could be seen as one sample of the inferred posterior distribution of the model. This is the first model that makes a concrete connection between spiking neural networks and the important new paradigm of sampling as an abstract description of the dynamics of the brain. The microcircuit that is developed could serve as a building block for larger models, which is already realized in a number of offspring papers that followed the first publication.

The idea for that unsupervised model-based approach, the connection to EM learning, the derivation of all formula work and proofs and the generalization to continuous time were done by me alone, based on the experience of the previous work. The extensive experimental work was done by Michael Pfeiffer, in collaboration with me. In the quest for the final proof based on stochastic approximation theory Lars Büsing was the resource of mathematical background and contributed the crucial final point in order to close the proof. The text was written mainly by myself with great contributions and overall reworking from Wolfgang Maass and Michael Pfeiffer.

The forth paper, in chapter 5, a joint work of Lars Büsing, Johannes Bill, Wolfgang Maass and me [27], is a seminal contribution to the community that lifts the initial idea of spikes as samples from the previous work to a new level and provides a concrete understanding of the dynamics of a recurrent spiking neural network as a Markov-Chain Monte Carlo (MCMC) sampling process. The paper gives a solid mathematical foundation of the description of the activity of the single neurons and derives a necessary condition on the functional relation between the membrane potential of a neuron and the activity of its neighbors in order for the whole network to be seen as a sampler of a well defined probability distribution. We show the astonishing capabilities of the network in an experiment that mimics the cortical processes in response to an ambiguous visual input stimulus. We are able to reproduce the well-known switching behavior between the two possible percepts underpinning the plausibility and strength of this concrete sampling-based modeling approach which was already hypothesized by [104], albeit without giving a concrete connection to the level of spiking neurons.

These results evolved from the quest initiated by Wolfgang Maass to generalize the previous results from SEM. The idea to replace the single shot one-spike-is-a-sample process by a sequentially chained sampler as well as the solid mathematical foundation of the final neural sampling theory was invented by Lars Büsing alone. The crucial perspective that this sampling process was powerful enough to sample specifically from Boltzmann distribution and in general from arbitrary probability distribution was developed by Lars Büsing and me together. The experiments were carried out by Johannes Bill, in close collaboration with Lars and me.

The fifth paper, in chapter 6, a joint work with Johannes Bill and Stefan Habenschuss [94], describes the valuable insight that neuronal homeostatic mechanisms can be understood as posterior constraints in the context of EM learning and it reveals

several beneficial implications of that connection. First it simplifies greatly the learning of probabilistic models in feedforward WTA-like network models. In that sense it completes the SEM architecture and delivers the proof that the learned probabilistic model can be a mixture of any distribution from the exponential family, e.g. multinomial, Bernoulli, Poisson, but also Gaussian or exponential distributions. More than that this extension abolishes the need for special input encodings through population codes as they were presented in the original SEM theory. The same theory of posterior constraints can also be applied to the sampling paradigm in order to reshape the posterior distribution in some favorable way. Apart from the advantages for the feedforward part the idea of posterior constraints can also be applied to the learning of recurrent weights between neurons, due to a second order homeostatic process. Even though there is not yet a biological evidence for such a process, the theory explains possible functional mechanisms that could explain the observed dependency between the distance of pyramidal neurons and the correlation of their output signals.

The basic idea of replacing the learning of prior probabilities in SEM by a regulation for uniform activities was initially tried out by Stefan Habenschuss. The elaboration and the conclusion that posterior constraints are a proper way for describing this setting was a joint result of all three authors. The analytic formulation and the generalization to the second order was done by Johannes Bill and Stefan Habenschuss in close and equal collaboration. The final paper was written by all three authors together.

2

Hebbian Learning of Bayes Optimal Decisions

Uncertainty is omnipresent when we perceive or interact with our environment, and the Bayesian framework provides computational methods for dealing with it. Mathematical models for Bayesian decision making typically require data-structures that are hard to implement in neural networks. This article shows that even the simplest and experimentally best supported type of synaptic plasticity, Hebbian learning, in combination with a sparse, redundant neural code, can in principle learn to infer optimal Bayesian decisions. We present a concrete Hebbian learning rule operating on log-probability ratios. Modulated by reward-signals, this Hebbian plasticity rule also provides a new perspective for understanding how Bayesian inference could support fast reinforcement learning in the brain. In particular we show that recent experimental results by Yang and Shadlen [232] on reinforcement learning of probabilistic inference in primates can be modeled in this way.

2.1 Introduction

Evolution is likely to favor those biological organisms which are able to maximize the chance of achieving correct decisions in response to multiple unreliable sources of evidence. Hence one may argue that probabilistic inference, rather than logical inference,

is the "mathematics of the mind", and that this perspective may help us to understand the principles of computation and learning in the brain [182]. Bayesian inference, or equivalently inference in Bayesian networks [22] is the most commonly considered framework for probabilistic inference, and a mathematical theory for learning in Bayesian networks has been developed.

Various attempts to relate these theoretically optimal models to experimentally supported models for computation and plasticity in networks of neurons in the brain have been made. [182] models Bayesian inference through an approximate implementation of the Belief Propagation algorithm (see [22]) in a network of spiking neurons. For reduced classes of probability distributions, [45, 46] proposed a method for spiking network models to learn Bayesian inference with an online approximation to an EM algorithm. The approach of [197] interprets the weight w_{ji} of a synaptic connection between neurons representing the random variables x_i and x_j as $\log \frac{p(x_i,x_j)}{p(x_i)\cdot p(x_j)}$, and presents algorithms for learning these weights.

Neural correlates of variables that are important for decision making under uncertainty had been presented e.g. in the recent experimental study by Yang and Shadlen [232]. In their study they found that firing rates of neurons in area LIP of macaque monkeys reflect the log-likelihood ratio (or log-odd) of the outcome of a binary decision, given visual evidence. The learning of such log-odds for Bayesian decision making can be reduced to learning weights for a linear classifier, given an appropriate but fixed transformation from the input to possibly nonlinear features [190]. We show that the optimal weights for the linear decision function are actually log-odds themselves, and the definition of the features determines the assumptions of the learner about statistical dependencies among inputs.

In this work we show that simple Hebbian learning [96] is sufficient to implement learning of Bayes optimal decisions for arbitrarily complex probability distributions. We present and analyze a concrete learning rule, which we call the *Bayesian Hebb rule*, and show that it provably converges towards correct log-odds. In combination with appropriate preprocessing networks this implements learning of different probabilistic decision making processes like e.g. Naive Bayesian classification. Finally we show that a reward-modulated version of this Hebbian learning rule can solve simple reinforcement learning tasks, and also provides a model for the experimental results of [232].

2.2 A Hebbian rule for learning log-odds

We consider the model of a linear threshold neuron with output y_0, where $y_0 = 1$ means that the neuron is firing and $y_0 = 0$ means non-firing. The neuron's current decision $\hat{y_0}$ whether to fire or not is given by a linear decision function $\hat{y_0} = \text{sign}(w_0 \cdot \mathit{constant} + \sum_{i=1}^{n} w_i y_i)$, where the y_i are the current firing states of all presynaptic neurons and w_i are the weights of the corresponding synapses.

We propose the following learning rule, which we call the Bayesian Hebb rule:

$$\boxed{\Delta w_i = \begin{cases} \eta\,(1 + e^{-w_i}), & \text{if } y_0 = 1 \text{ and } y_i = 1 \\ -\eta\,(1 + e^{w_i}), & \text{if } y_0 = 0 \text{ and } y_i = 1 \\ 0, & \text{if } y_i = 0. \end{cases}} \tag{2.1}$$

This learning rule is purely local, i.e. it depends only on the binary firing state of the pre- and postsynaptic neuron y_i and y_0, the current weight w_i and a learning rate η. Under the assumption of a stationary joint probability distribution of the pre- and postsynaptic firing states $y_0, y_1, \ldots, y_n$ the Bayesian Hebb rule learns log-probability ratios of the postsynaptic firing state y_0, conditioned on a corresponding presynaptic firing state y_i. We consider in this article the use of the rule in a supervised, teacher forced mode (see Section 2.3), and also in a reinforcement learning mode (see Section 2.4). We will prove that the rule converges globally to the target weight value w_i^*, given by

$$w_i^* = \log \frac{p(y_0 = 1 | y_i = 1)}{p(y_0 = 0 | y_i = 1)} \quad . \tag{2.2}$$

We first show that the expected update $\mathrm{E}[\Delta w_i]$ under (2.1) vanishes at the target value w_i^*:

$$\begin{aligned} \mathrm{E}[\Delta w_i^*] = 0 \;&\Leftrightarrow\; p(y_0{=}1, y_i{=}1)\eta(1 + e^{-w_i^*}) - p(y_0{=}0, y_i{=}1)\eta(1 + e^{w_i^*}) = 0 \\ &\Leftrightarrow\quad \frac{1 + e^{w_i^*}}{1 + e^{-w_i^*}} = \frac{p(y_0{=}1, y_i{=}1)}{p(y_0{=}0, y_i{=}1)} \\ &\Leftrightarrow\quad w_i^* = \log \frac{p(y_0{=}1 | y_i{=}1)}{p(y_0{=}0 | y_i{=}1)} \quad . \end{aligned} \tag{2.3}$$

Since the above is a chain of equivalence transformations, this proves that w_i^* is the only equilibrium value of the rule. The weight vector $\mathbf{w}^*$ is thus a global point-attractor with regard to expected weight changes of the Bayesian Hebb rule (2.1) in the n-dimensional weight-space $\mathbb{R}^n$.

Furthermore we show, using the result from (2.3), that the expected weight change at any current value of w_i points in the direction of w_i^*. Consider some arbitrary intermediate weight value $w_i = w_i^* + 2\epsilon$:

$$\begin{aligned} \mathrm{E}[\Delta w_i]|_{w_i^*+2\epsilon} &= \mathrm{E}[\Delta w_i]|_{w_i^*+2\epsilon} - E[\Delta w_i]|_{w_i^*} \\ &\propto p(y_0{=}1, y_i{=}1)e^{-w_i^*}(e^{-2\epsilon}-1) - p(y_0{=}0, y_i{=}1)e^{w_i^*}(e^{2\epsilon}-1) \\ &= (p(y_0{=}0, y_i{=}1)e^{-\epsilon} + p(y_0{=}1, y_i{=}1)e^{\epsilon})(e^{-\epsilon}-e^{\epsilon}) \quad . \end{aligned} \tag{2.4}$$

The first factor in (2.4) is always non-negative, hence $\epsilon < 0$ implies $E[\Delta w_i] > 0$, and $\epsilon > 0$ implies $E[\Delta w_i] < 0$. The Bayesian Hebb rule is therefore always expected to perform updates in the right direction, and the initial weight values or perturbations of the weights decay exponentially fast.

Already after having seen a finite set of examples $\langle y_0, \ldots, y_n \rangle \in \{0,1\}^{n+1}$, the Bayesian Hebb rule closely approximates the optimal weight vector $\hat{\mathbf{w}}$ that can be inferred from the data. A traditional frequentist's approach would use counters $a_i = \#[y_0{=}1 \wedge y_i{=}1]$ and $b_i = \#[y_0{=}0 \wedge y_i{=}1]$ to estimate every w_i^* by

$$\hat{w}_i = \log \frac{a_i}{b_i} \ . \tag{2.5}$$

A Bayesian approach would model $p(y_0|y_i)$ with an (initially flat) *Beta*-distribution, and use the counters a_i and b_i to update this belief [22], leading to the same MAP estimate $\hat{w}_i$. Consequently, in both approaches a new example with $y_0 = 1$ and $y_i = 1$ leads to the update

$$\hat{w}_i^{new} = \log \frac{a_i + 1}{b_i} = \log \frac{a_i}{b_i}\left(1 + \frac{1}{a_i}\right) = \hat{w}_i + \log(1 + \frac{1}{N_i}(1 + e^{-\hat{w}_i})) \ , \tag{2.6}$$

where $N_i := a_i + b_i$ is the number of previously processed examples with $y_i = 1$, thus $\frac{1}{a_i} = \frac{1}{N_i}(1 + \frac{b_i}{a_i})$. Analogously, a new example with $y_0 = 0$ and $y_i = 1$ gives rise to the update

$$\hat{w}_i^{new} = \log \frac{a_i}{b_i + 1} = \log \frac{a_i}{b_i}\left(\frac{1}{1 + \frac{1}{b_i}}\right) = \hat{w}_i - \log(1 + \frac{1}{N_i}(1 + e^{\hat{w}_i})). \tag{2.7}$$

Furthermore, $\hat{w}_i^{new} = \hat{w}_i$ for a new example with $y_i = 0$. Using the approximation $\log(1+\alpha) \approx \alpha$ the update rules (2.6) and (2.7) yield the Bayesian Hebb rule (2.1) with an adaptive learning rate $\eta_i = \frac{1}{N_i}$ for each synapse.

In fact, a result of Robbins-Monro (see [17] for a review) implies that the updating of weight estimates $\hat{w}_i$ according to (2.6) and (2.7) converges to the target values w_i^* not only for the particular choice $\eta_i^{(N_i)} = \frac{1}{N_i}$, but for any sequence $\eta_i^{(N_i)}$ that satisfies $\sum_{N_i=1}^{\infty} \eta_i^{(N_i)} = \infty$ and $\sum_{N_i=1}^{\infty} (\eta_i^{(N_i)})^2 < \infty$. More than that the Supermartingale Convergence Theorem (see [17]) guarantees convergence in distribution even for a sufficiently small constant learning rate.

Learning rate adaptation

One can see from the above considerations that the Bayesian Hebb rule with a constant learning rate η converges globally to the desired log-odds. A too small constant learning rate, however, tends to slow down the initial convergence of the weight vector, and a too large constant learning rate produces larger fluctuations once the steady state is reached.

(2.6) and (2.7) suggest a decaying learning rate $\eta_i^{(N_i)} = \frac{1}{N_i}$, where N_i is the number of preceding examples with $y_i = 1$. We will present a learning rate adaptation mechanism that avoids biologically implausible counters, and is robust enough to deal even with non-stationary distributions.

Since the Bayesian Hebb rule and the Bayesian approach of updating $Beta$-distributions for conditional probabilities are closely related, it is reasonable to expect that the distribution of weights w_i over longer time periods with a non-vanishing learning rate will resemble a $Beta(a_i, b_i)$-distribution transformed to the log-odd domain. The parameters a_i and b_i in this case are not exact counters anymore but correspond to virtual sample sizes, depending on the current learning rate. We formalize this statistical model of w_i by

$$\sigma(w_i) = \frac{1}{1 + e^{-w_i}} \sim Beta(a_i, b_i) \iff w_i \sim \frac{\Gamma(a_i + b_i)}{\Gamma(a_i)\Gamma(b_i)} \sigma(w_i)^{a_i} \sigma(-w_i)^{b_i},$$

In practice this model turned out to capture quite well the actually observed quasi-stationary distribution of w_i. It can be shown analytically that $\mathrm{E}[w_i] \approx \log \frac{a_i}{b_i}$ and $\mathrm{Var}[w_i] \approx \frac{1}{a_i} + \frac{1}{b_i}$. A learning rate adaptation mechanism at the synapse that keeps track of the observed mean and variance of the synaptic weight can therefore recover estimates of the virtual sample sizes a_i and b_i. The following mechanism, which we call

variance tracking implements this by computing running averages of the weights and the squares of weights in $\bar{w}_i$ and $\bar{q}_i$:

$$\boxed{\begin{array}{lcl} \eta_i^{new} & \leftarrow & \frac{\bar{q}_i - \bar{w}_i^2}{1+\cosh \bar{w}_i} \\ \bar{w}_i^{new} & \leftarrow & (1-\eta_i)\,\bar{w}_i + \eta_i\, w_i \\ \bar{q}_i^{new} & \leftarrow & (1-\eta_i)\,\bar{q}_i + \eta_i\, w_i^2 \quad . \end{array}} \tag{2.8}$$

In practice this mechanism decays like $\frac{1}{N_i}$ under stationary conditions, but is also able to handle changing input distributions. It was used in all presented experiments for the Bayesian Hebb rule.

2.3 Hebbian learning of Bayesian decisions

We now show how the Bayesian Hebb rule can be used to learn Bayes optimal decisions. The first application is the Naive Bayesian classifier, where a binary target variable x_0 should be inferred from a vector of multinomial variables $\mathbf{x} = \langle x_1, \ldots, x_m \rangle$, under the assumption that the x_i's are conditionally independent given x_0, thus $p(x_0, \mathbf{x}) = p(x_0) \prod_1^m p(x_k|x_0)$. Using basic rules of probability theory the posterior probability ratio for $x_0 = 1$ and $x_0 = 0$ can be derived:

$$\begin{aligned} \frac{p(x_0{=}1|\mathbf{x})}{p(x_0{=}0|\mathbf{x})} = \frac{p(x_0{=}1)}{p(x_0{=}0)} \prod_{k=1}^{m} \frac{p(x_k|x_0{=}1)}{p(x_k|x_0{=}0)} = \left(\frac{p(x_0{=}1)}{p(x_0{=}0)}\right)^{(1-m)} \prod_{k=1}^{m} \frac{p(x_0{=}1|x_k)}{p(x_0{=}0|x_k)} = \\ = \left(\frac{p(x_0{=}1)}{p(x_0{=}0)}\right)^{(1-m)} \prod_{k=1}^{m} \prod_{j=1}^{m_k} \left(\frac{p(x_0{=}1|x_k{=}j)}{p(x_0{=}0|x_k{=}j)}\right)^{I(x_k=j)} , \end{aligned} \tag{2.9}$$

where m_k is the number of different possible values of the input variable x_k, and the indicator function I is defined as $I(true) = 1$ and $I(false) = 0$.

Let the m input variables $x_1, \ldots, x_m$ be represented through the binary firing states $y_1, \ldots, y_n \in \{0, 1\}$ of the n presynaptic neurons in a population coding manner. More precisely, let each input variable $x_k \in \{1, \ldots, m_k\}$ be represented by m_k neurons, where each neuron fires only for one of the m_k possible values of x_k. Formally we define the simple preprocessing (SP)

$$\mathbf{y} = \left[\boldsymbol{\phi}(x_1)^\mathsf{T}, \ldots, \boldsymbol{\phi}(x_m)^\mathsf{T}\right]^\mathsf{T} \quad \text{with} \quad \boldsymbol{\phi}(x_k)^\mathsf{T} = [I(x_k = 1), \ldots, I(x_k = m_k)] \ . \tag{2.10}$$

The binary target variable x_0 is represented directly by the binary state y_0 of the postsynaptic neuron. Substituting the state variables $y_0, y_1, \ldots, y_n$ in (2.9) and taking

the logarithm leads to

$$\log \frac{p(y_0 = 1|\mathbf{y})}{p(y_0 = 0|\mathbf{y})} = (1-m) \log \frac{p(y_0 = 1)}{p(y_0 = 0)} + \sum_{i=1}^{n} y_i \log \frac{p(y_i = 1|y_0 = 1)}{p(y_i = 1|y_0 = 0)} \ .$$

Hence the optimal decision under the Naive Bayes assumption is

$$\hat{y_0} = \text{sign}((1-m)w_0^* + \sum_{i=1}^{n} w_i^* \ y_i) \quad .$$

The optimal weights w_0^* and w_i^*

$$w_0^* = \log \frac{p(y_0 = 1)}{p(y_0 = 0)} \qquad \text{and} \qquad w_i^* = \log \frac{p(y_0 = 1|y_i = 1)}{p(y_0 = 0|y_i = 1)} \qquad \text{for} \qquad i = 1, \ldots, n.$$

are obviously log-odds which can be learned by the Bayesian Hebb rule (the bias weight w_0 is simply learned as an unconditional log-odd).

2.3.1 Learning Bayesian decisions for arbitrary distributions

We now address the more general case, where conditional independence of the input variables $x_1, \ldots, x_m$ cannot be assumed. In this case the dependency structure of the underlying distribution is given in terms of an arbitrary Bayesian network BN for discrete variables (see e.g. Figure 2.1 A). Without loss of generality we choose a numbering scheme of the nodes of the BN such that the node to be learned is x_0 and its direct children are $x_1, \ldots, x_{m'}$. This implies that the BN can be described by $m+1$ (possibly empty) parent sets defined by

$$\mathtt{P}_k \quad = \quad \{i \mid \text{a directed edge } x_i \to x_k \text{ exists in BN} \ \text{ and } \ i \geq 1\} \quad .$$

The joint probability distribution on the variables $x_0, \ldots, x_m$ in BN can then be factored and evaluated for $x_0 = 1$ and $x_0 = 0$ in order to obtain the probability ratio

$$\frac{p(x_0 = 1, \mathbf{x})}{p(x_0 = 0, \mathbf{x})} = \frac{p(x_0 = 1|\mathbf{x})}{p(x_0 = 0|\mathbf{x})} = \frac{p(x_0 = 1|x_{\mathtt{P}_0})}{p(x_0 = 0|x_{\mathtt{P}_0})} \prod_{k=1}^{m'} \frac{p(x_k|\mathbf{x}_{\mathtt{P}_k}, x_0 = 1)}{p(x_k|\mathbf{x}_{\mathtt{P}_k}, x_0 = 0)} \prod_{k=m'+1}^{m} \frac{p(x_k|\mathbf{x}_{\mathtt{P}_k})}{p(x_k|\mathbf{x}_{\mathtt{P}_k})} \quad .$$

Obviously, the last term cancels out, and by applying Bayes' rule and taking the logarithm the target log-odd can be expressed as a sum of conditional log-odds only:

$$\log \frac{p(x_0{=}1|\mathbf{x})}{p(x_0{=}0|\mathbf{x})} = \log \frac{p(x_0{=}1|\mathbf{x}_{\mathtt{P}_0})}{p(x_0{=}0|\mathbf{x}_{\mathtt{P}_0})} + \sum_{k=1}^{m'} \left(\log \frac{p(x_0{=}1|x_k, \mathbf{x}_{\mathtt{P}_k})}{p(x_0{=}0|x_k, \mathbf{x}_{\mathtt{P}_k})} - \log \frac{p(x_0{=}1|\mathbf{x}_{\mathtt{P}_k})}{p(x_0{=}0|\mathbf{x}_{\mathtt{P}_k})} \right). \tag{2.11}$$

We now develop a suitable sparse encoding of of $x_1, \ldots, x_m$ into binary variables $y_1, \ldots, y_n$ (with $n \gg m$) such that the decision function (2.11) can be written as a weighted sum, and the weights correspond to conditional log-odds of y_i's. Figure 2.1 B illustrates such a sparse code: One binary variable is created for every possible value assignment to a variable and all its parents, and one additional binary variable is created for every possible value assignment to the parent nodes only. Formally, the previously introduced population coding operator ϕ is generalized such that $\phi(x_{i_1}, x_{i_2}, \ldots, x_{i_l})$ creates a vector of length $\prod_{j=1}^{l} m_{i_j}$ that equals zero in all entries except for one 1-entry which identifies by its position in the vector the present assignment of the input variables $x_{i_1}, \ldots, x_{i_l}$. The concatenation of all these population coded groups is collected in the vector $\mathbf{y}$ of length n

$$\mathbf{y} = \left[\phi(\mathbf{x}_{\mathrm{P}_0})^\mathsf{T}, \phi(x_1, \mathbf{x}_{\mathrm{P}_1})^\mathsf{T}, -\phi(\mathbf{x}_{\mathrm{P}_1})^\mathsf{T}, \ldots, \phi(x_m, \mathbf{x}_{\mathrm{P}_m})^\mathsf{T}, -\phi(\mathbf{x}_{\mathrm{P}_m})^\mathsf{T}\right]^\mathsf{T} \quad . \tag{2.12}$$

The negated vector parts in (2.12) correspond to the negative coefficients in the sum in (2.11). Inserting the sparse coding (2.12) into (2.11) allows writing the Bayes optimal decision function (2.11) as a pure sum of log-odds of the target variable:

$$\hat{x_0} = \hat{y_0} = \mathrm{sign}(\sum_{i=1}^{n} w_i^* y_i), \qquad \text{with} \qquad w_i^* = \log \frac{p(y_0{=}1|y_i{\neq}0)}{p(y_0{=}0|y_i{\neq}0)} \quad .$$

Every synaptic weight w_i can be learned efficiently by the Bayesian Hebb rule (2.1) with the formal modification that the update is not only triggered by $y_i{=}1$ but in general whenever $y_i{\neq}0$ (which obviously does not change the behavior of the learning process). A neuron that learns with the Bayesian Hebb rule on inputs that are generated by the generalized preprocessing (*GP*) defined in (2.12) therefore approximates the Bayes optimal decision function (2.11), and converges quite fast to the best performance that any probabilistic inference could possibly achieve (see Figure 2.2B).

2.4 The Bayesian Hebb rule in reinforcement learning

We show in this section that a reward-modulated version of the Bayesian Hebb rule enables a learning agent to solve simple reinforcement learning tasks. We consider the standard operant conditioning scenario, where the learner receives at each trial an input $\mathbf{x} = \langle x_1, \ldots, x_m \rangle$, chooses an action α out of a set of possible actions A, and receives a binary reward signal $r \in \{0, 1\}$ with probability $p(r|\mathbf{x}, a)$. The learner's goal is to learn

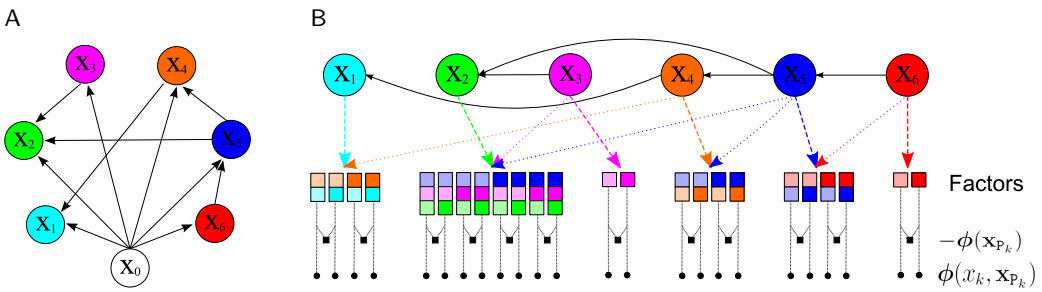

Figure 2.1: A) An example Bayesian network with general connectivity. **B)** Population coding applied to the Bayesian network shown in panel A. For each combination of values of the variables $\{x_k, \mathbf{x}_{\mathbf{P}_k}\}$ of a factor there is exactly one neuron (indicated by a black circle) associated with the factor that outputs the value 1. In addition OR's of these values are computed (black squares). We refer to the resulting preprocessing circuit as generalized preprocessing (GP).

(as fast as possible) a policy $\pi(\mathbf{x}, a)$ so that action selection according to this policy maximizes the average reward. In contrast to the previous learning tasks, the learner has to explore different actions for the same input to learn the reward-probabilities for all possible actions. The agent might for example choose actions stochastically with $\pi(\mathbf{x}, a = \alpha) = p(r = 1|\mathbf{x}, a = \alpha)$, which corresponds to the *matching behavior* phenomenon often observed in biology [213]. This policy was used during training in our computer experiments.

The goal is to infer the probability of binary reward, so it suffices to learn the log-odds $log\frac{p(r=1|\mathbf{x},a)}{p(r=0|\mathbf{x},a)}$ for every action, and choose the action that is most likely to yield reward (e.g. by a Winner-Take-All structure). If the reward probability for an action $a = \alpha$ is defined by some Bayesian network BN, one can rewrite this log-odd as

$$\log \frac{p(r=1|\mathbf{x}, a=\alpha)}{p(r=0|\mathbf{x}, a=\alpha)} = \log \frac{p(r=1|a=\alpha)}{p(r=0|a=\alpha)} + \sum_{k=1}^{m} \log \frac{p(x_k|\mathbf{x}_{\mathbf{P}_k}, r=1, a=\alpha)}{p(x_k|\mathbf{x}_{\mathbf{P}_k}, r=0, a=\alpha)}. \quad (2.13)$$

In order to use the Bayesian Hebb rule, the input vector $\mathbf{x}$ is preprocessed to obtain a binary vector $\mathbf{y}$. Both a simple population code such as (2.10), or generalized preprocessing as in (2.12) and Figure 2.1B can be used, depending on the assumed dependency structure. The reward log-odd (2.13) for the preprocessed input vector $\mathbf{y}$ can then be written as a linear sum

$$\log \frac{p(r=1|\mathbf{y}, a=\alpha)}{p(r=0|\mathbf{y}, a=\alpha)} \quad = \quad w^*_{\alpha,0} + \sum_{i=1}^{n} w^*_{\alpha,i}\, y_i \quad ,$$

where the optimal weights are $w^*_{\alpha,0} = \log \frac{p(r=1|a=\alpha)}{p(r=0|a=\alpha)}$ and $w^*_{\alpha,i} = \log \frac{p(r=1|y_i\neq 0,a=\alpha)}{p(r=0|y_i\neq 0,a=\alpha)}$. These log-odds can be learned for each possible action α with a reward-modulated version of the Bayesian Hebb rule (2.1):

$$\boxed{\Delta w_{\alpha,i} = \begin{cases} \eta \cdot (1 + e^{-w_{\alpha,i}}), & \text{if } r = 1, y_i \neq 0, a = \alpha \\ -\eta \cdot (1 + e^{w_{\alpha,i}}), & \text{if } r = 0, y_i \neq 0, a = \alpha \\ 0, & \text{otherwise} \end{cases}} \tag{2.14}$$

The attractive theoretical properties of the Bayesian Hebb rule for the prediction case apply also to the case of reinforcement learning. The weights corresponding to the optimal policy are the only equilibria under the reward-modulated Bayesian Hebb rule, and are also global attractors in weight space, independently of the exploration policy.

2.5 Experimental Results

2.5.1 Results for prediction tasks

We have tested the Bayesian Hebb rule on 400 different prediction tasks, each of them defined by a general (non-Naive) Bayesian network of 7 binary variables. The networks were randomly generated by the algorithm of [106]. From each network we sampled 2000 training and 5000 test examples, and measured the percentage of correct predictions after every update step.

The performance of the predictor was compared to the Bayes optimal predictor, and to online logistic regression, which fits a linear model by gradient descent on the cross-entropy error function. This non-Hebbian learning approach is in general the best performing online learning approach for linear discriminators [22]. Figure 2.2A shows that the Bayesian Hebb rule with the simple preprocessing (2.10) generalizes better from a few training examples, but is outperformed by logistic regression in the long run, since the Naive Bayes assumption is not met. With the generalized preprocessing (2.12), the Bayesian Hebb rule learns fast and converges to the Bayes optimum (see Figure 2.2B). In Figure 2.2C we show that the Bayesian Hebb rule is robust to noisy updates - a condition very likely to occur in biological systems. We modified the weight update Δw_i such that it was uniformly distributed in the interval $\Delta w_i \pm \gamma\%$. Even such imprecise implementations of the Bayesian Hebb rule perform very well. Similar results can be obtained if the exp-function in (2.1) is replaced by a low-order Taylor approximation.

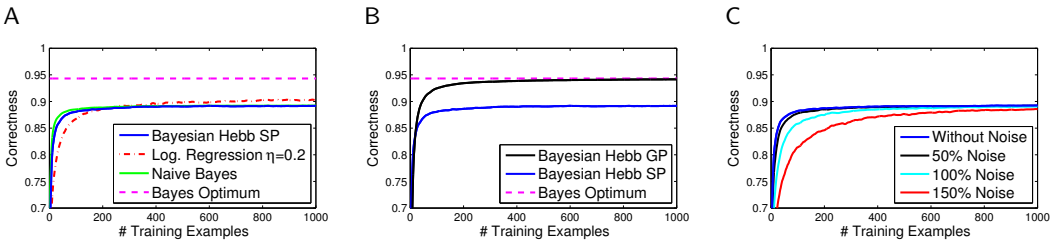

Figure 2.2: Performance comparison for prediction tasks. **A)** The Bayesian Hebb rule with simple preprocessing (*SP*) learns as fast as Naive Bayes, and faster than logistic regression (with optimized constant learning rate). **B)** The Bayesian Hebb rule with generalized preprocessing (*GP*) learns fast and converges to the Bayes optimal prediction performance. **C)** Even a very imprecise implementation of the Bayesian Hebb rule (noisy updates, uniformly distributed in $\Delta w_i \pm \gamma\%$) yields almost the same learning performance.

2.5.2 Results for action selection tasks

The reward-modulated version (2.14), of the Bayesian Hebb rule was tested on 250 random action selection tasks with $m = 6$ binary input attributes, and 4 possible actions. For every action a random Bayesian network [106] was drawn to model the input and reward distributions. The agent received stochastic binary rewards for every chosen action, updated the weights $w_{\alpha,i}$ according to (2.14), and measured the average reward on 500 independent test trials.

In Figure 2.3A we compare the reward-modulated Bayesian Hebb rule with simple population coding (2.10) (*Bayesian Hebb SP*), and generalized preprocessing (2.12) (*Bayesian Hebb GP*), to the standard learning model for simple conditioning tasks, the non-Hebbian Rescorla-Wagner rule [184]. The reward-modulated Bayesian Hebb rule learns as fast as the Rescorla-Wagner rule, and achieves in combination with generalized preprocessing a higher performance level. The widely used tabular Q-learning algorithm, in comparison is slower than the other algorithms, since it does not generalize, but it converges to the optimal policy in the long run.

2.5.3 A model for the experiment of Yang and Shadlen

In the experiment by Yang and Shadlen [232], a monkey had to choose between gazing towards a red target R or a green target G. The probability that a reward was received at either choice depended on four visual input stimuli that had been shown at the

beginning of the trial. Every stimulus was one shape out of a set of ten possibilities and had an associated weight, which had been defined by the experimenter. The sum of the four weights yielded the log-odd of obtaining a reward at the red target, and a reward for each trial was assigned accordingly to one of the targets. The monkey thus had to combine the evidence from four visual stimuli to optimize its action selection behavior.

In the model of the task it is sufficient to learn weights only for the action $a = R$, and select this action whenever the log-odd using the current weights is positive, and G otherwise. A simple population code as in (2.10) encoded the 4-dimensional visual stimulus into a 40-dimensional binary vector $\mathbf{y}$. In our experiments, the reward-modulated Bayesian Hebb rule learns this task as fast and with similar quality as the non-Hebbian Rescorla-Wagner rule. Furthermore Figures 2.3B, C show that it produces after learning similar behavior as that reported for two monkeys in [232].

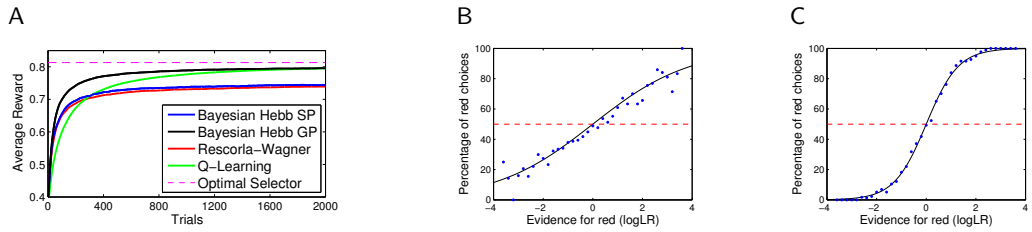

Figure 2.3: A) On 250 4-action conditioning tasks with stochastic rewards, the reward-modulated Bayesian Hebb rule with simple preprocessing (SP) learns similarly as the Rescorla-Wagner rule, and substantially faster than Q-learning. With generalized preprocessing (GP), the rule converges to the optimal action-selection policy. **B, C)** Action selection policies learned by the reward-modulated Bayesian Hebb rule in the task by Yang and Shadlen [232] after 100 (B), and 1000 (C) trials are qualitatively similar to the policies adopted by monkeys H and J in [232] after learning.

2.6 Discussion

We have shown that the simplest and experimentally best supported local learning mechanism, Hebbian learning, is sufficient to learn Bayes optimal decisions. We have introduced and analyzed the Bayesian Hebb rule, a training method for synaptic weights, which converges fast and robustly to optimal log-probability ratios, without requiring

any communication between plasticity mechanisms for different synapses. We have shown how the same plasticity mechanism can learn Bayes optimal decisions under different statistical independence assumptions, if it is provided with an appropriately preprocessed input. We have demonstrated on a variety of prediction tasks that the Bayesian Hebb rule learns very fast, and with an appropriate sparse preprocessing mechanism for groups of statistically dependent features its performance converges to the Bayes optimum. Our approach therefore suggests that sparse, redundant codes of input features may simplify synaptic learning processes in spite of strong statistical dependencies. Finally we have shown that Hebbian learning also suffices for simple instances of reinforcement learning. The Bayesian Hebb rule, modulated by a signal related to rewards, enables fast learning of optimal action selection. Experimental results of [232] on reinforcement learning of probabilistic inference in primates can be partially modeled in this way with regard to resulting behaviors.

An attractive feature of the Bayesian Hebb rule is its ability to deal with the addition or removal of input features through the creation or deletion of synaptic connections, since no relearning of weights is required for the other synapses. In contrast to discriminative neural learning rules, our approach is generative, which according to [160] leads to faster generalization. Therefore the learning rule may be viewed as a potential building block for models of the brain as a self-organizing and fast adapting probabilistic inference machine.

3

Reward-modulated Hebbian Learning of Decision Making

We introduce a framework for decision making in which the learning of decision making is reduced to its simplest and biologically most plausible form: Hebbian learning on a linear neuron. We cast our Bayesian-Hebb learning rule as reinforcement learning in which certain decisions are rewarded, and prove that each synaptic weight will on average converge exponentially fast to the log-odd of receiving a reward when its pre- and post-synaptic neurons are active. In our simple architecture, a particular action is selected from the set of candidate actions by a winner-take-all operation. The global reward assigned to this action then modulates the update of each synapse. Apart from this global reward signal our reward-modulated Bayesian Hebb rule is a pure Hebb update that depends only on the co-activation of the pre- and postsynaptic neurons, and not on the weighted sum of all presynaptic inputs to the post-synaptic neuron as in the perceptron learning rule or the Rescorla-Wagner rule. This simple approach to action-selection learning requires that information about sensory inputs be presented to the Bayesian decision stage in a suitably pre-processed form resulting from other adaptive processes (acting on a larger time scale) that detect salient dependencies among input features. Hence our proposed framework for fast learning of decisions also provides interesting new hypotheses regarding neural nodes and computational goals of cortical areas

that provide input to the final decision stage.

3.1 Introduction

A typical decision making task of an organism requires the evaluation of multiple alternative actions, with the goal of maximizing the probability of obtaining positive reward. If input signals provide only uncertain cues, and reward is obtained stochastically in response to actions, then Bayesian statistics provides a mathematical framework for the optimal integration of all available information. Bayes' theorem can be used to calculate the probability that an action yields a reward, given the current sensory input and the current internal state of an organism. The goal of this article is to present the simplest possible neural network model that can make such an evaluation, where simplicity is assessed both in terms of computational operations, and the complexity of the learning method.

A large number of experimental results suggest that animals do indeed make decisions based on Bayesian integration of information about stimulus-action-reward contingencies. For example, [213] (see [214] for a review) have shown that monkeys use the *matching behavior strategy*, in which the frequency with which a particular action is chosen matches the expected reward for that action. [232] have shown that the previous experience of macaque monkeys in probabilistic decision tasks is represented by the firing rates of neurons in area LIP in the form of the log-likelihood ratio (or log-odd) of receiving a reward for a particular action a in response to a stimulus $\mathbf{x}$ (in an experiment where the monkey received in each trial either no reward, or a reward of unit size, depending on the choice of the monkey among two possible actions).

We show that an optimal action selection policy can be reduced to a Winner-Take-All (WTA) operation applied to linear gates, which receive suitably preprocessed inputs (see Figure 3.1). Furthermore, we show that the updating of the WTA circuit in the face of new evidence can be reduced to the application of a local reward-modulated Hebbian learning rule to each linear gate. We call this rule the Bayesian Hebb Rule. Despite the simplicity of this model, one can prove that it enables fast learning of near optimal decision making, which is remarkable because rigorous insight into convergence properties of Hebbian learning rules is often lacking.

Figure 3.1: Winner-Take-All (WTA) architecture for learning of decision making. First, the multinomial input variables $x_1, \ldots, x_m$ are preprocessed by a fixed circuit (which implements some type of population coding) to yield binary variables $y_1, \ldots, y_n$. For every possible action a there is an associated linear neuron L_a which computes a weighted sum $\sum_{i=0}^{n} w_{a,i} y_i$ of the variables $y_1, \ldots, y_n$. The neuron L_a with the largest weighted sum "wins", i.e. $z_a = 1$, and action a is selected.

WTA (see [235] for a review) is a very simple computational operation that selects the largest among l values $L_1, \ldots, L_l$. This selection is usually encoded through l binary outputs $z_1, \ldots, z_l$, where $z_a = 1$ if L_a is selected as the largest input (ties can be broken arbitrarily), else $z_a = 0$ (see Figure 3.1). In an action selection framework this output then triggers the selection of the a^{th} among l possible actions. Each value L_a is just a weighted sum

$$L_a = \sum_{i=0}^{n} w_{a,i} \; y_i$$

of variables $y_1, \ldots, y_n$ (and a dummy variable $y_0 \equiv 1$ that allows to use $w_{a,0}$ as a bias). Despite its simplicity, the resulting WTA-circuit is computationally quite powerful [144].

The main contribution of this article is a novel learning algorithm for the weights $w_{a,i}$ of the linear gates L_a. We show that for a suitable fixed preprocessing (that transfers the original input variables x_k into binary variables y_i) the optimal value $w^*_{a,i}$ for the weight $w_{a,i}$ in Figure 3.1 is the log-likelihood ratio (or log-odd) of receiving a reward for a particular action a, provided that the binary feature y_i is activated by the preprocessing function, i.e.

$$w^*_{a,i} = \log \frac{p(r = 1 | y_i = 1, a)}{p(r = 0 | y_i = 1, a)} \quad . \tag{3.1}$$

In the asymptotic case, where all weights $w_{a,i}$ have converged to their respective target values $w^*_{a_i}$, the policy of the WTA-circuit in Figure 3.1 is optimal in the sense that for any input signal the action with the highest chance to deliver reward is chosen. We also show that after finitely many training trials steps the weights closely approximate the optimal weights that can be inferred from the previously observed data.

Our algorithm for reward-modulated learning of optimal weights uses only Hebbian learning, a form of learning for which there is strong experimental evidence [1, 29, 66]. [96] proposed (see [66] for a recent review) that a synapse from neuron A to neuron B is strengthened if A and B often fire together. But several studies have shown that Hebbian synaptic plasticity requires a third signal (often in the form of neuromodulators) in order to consolidate weight changes [14, 59, 137, 185]. It is often assumed that the third signal provides information about reward or reward expectations. Hence learning rules involving these signals are referred to as reward-modulated learning rules.

Hebbian learning, such as in the proposed Bayesian Hebb rule should be contrasted with non-Hebbian learning rules such as the perceptron learning rule (also referred to as Delta-rule), or the Rescorla-Wagner rule [184], which are harder to support on the basis of experimental data for synaptic plasticity. In these latter learning rules the change Δw_i of a synaptic weight w_i at a single synapse depends not only on the current activation values of the pre- and postsynaptic neuron and the current value of w_i (and possibly a reward-related third signal), but also on the current values of the other weights and the activation values of all other neurons that provide synaptic input to the same postsynaptic neuron (more precisely: on the value of the weighted sum of all presynaptic inputs).

We present a mechanism for reward-modulated local learning of the weights $w_{a,i}$ that permits them to converge (on average) to the ideal value (3.1). Learning from rewards is conceptionally different from learning with a supervisor that informs the learner about the correct choice. In reward-based learning, the learner must explore different actions multiple times, even if he assumes that other actions would be better in the given situation. This strategy is necessary to avoid premature convergence to suboptimal policies.

We want to make clear that in this article we do not study the learning of sequences of actions as in general reinforcement learning [217], but investigate scenarios like in operant conditioning, where decisions have to be made based on learned immediate reward probabilities for single actions. We follow however the terminology proposed for example in [38], and subsume the latter also under the term reinforcement learning.

We will provide in this article a rigorous theoretical analysis of the convergence properties of the Bayesian Hebb rule. Because our learning rule makes online updates after every training trial, rather than performing a batch update after collecting a set

of data, we are interested in the asymptotic behavior of the rule, as well as its online performance. Non-Hebbian learning rules usually perform gradient descent optimization along an error surface. If local minima exist on the error surface, this approach always carries the risk of becoming trapped in suboptimal solutions, from which it cannot escape. In contrast, the optimal values of the weights to be learned by the Bayesian Hebb rule act as global fixed point attractors in weight-space with regard to expected weight updates of the Bayesian Hebb rule. Our analysis shows that the weights learned during training are very close to the optimal values that can be inferred from finitely many training trials, and they converge exponentially fast to the optimal values. We will also demonstrate that an extremely simple linear approximation to the Bayesian Hebb rule performs almost equally well.

Bayesian decision making combines information from many variables, and therefore must consider statistical dependencies amongst them. An influential paper by [190] noted that decision making can be reduced to the computation of weighted sums, provided that the input signals are properly pre-processed (see also [51]). This observation motivates our use of the neural network model shown in Figure 3.1. [190] proved his results in the context of linear statistical queries for probabilistic classification. We now extend this approach to the case of policy learning by incorporating a WTA gate for action selection. [190] noted that the set of features produced by the preprocessing function must be related to independence assumptions among input variables. We show that these features correspond to the factors in a factor graph [130] of the input- and reward distribution.

One particularly simple case is *Naive Bayes*, which assumes that all input variables are conditionally independent given one particular target variable, e.g. the occurrence of reward. In this case it is sufficient to know the reward-prediction probabilities for every input variable and every action separately, since then the reward probability given the complete input is the product of all individual predictors. We provide a simple preprocessing function for this case, which does not use any information about statistical dependencies of input variables, but leads to satisfactory policies.

The general case, in which there are statistical dependencies among input variables, requires more complex algorithms for Bayesian inference. Graphical models like Bayesian networks [22] and factor graphs [130] are used to model conditional dependencies among variables, and inference algorithms operate by passing messages along edges

of the graphs. Factor graphs are particularly useful tools. They consider groups of dependent variables as *factor nodes*, in which functions of all connected variable nodes are computed. Inference in these models is performed using the sum-product algorithm [22, 130], which is conceptually simpler than the belief propagation algorithms used for inference in general Bayesian networks. Recent work [211] has shown that these factor nodes can be implemented in networks of spiking neurons. In this article we define an optimal generalized preprocessing function based on the factor graph representation of the reward distribution. This provides a concrete processing goal for multimodal integration in sensory areas, and links the theory of factor graphs to experimentally observed neural population codes. These codes, as all other components of our framework, are easily implemented in neural networks, and allow fast and robust learning with the Hebbian learning algorithms presented in this article.

We assume here that the graph structure of the underlying Bayesian network is known, but not the parameters of it (i.e., the probability distribution). We do not address the problem of structure learning, which is a very different task, and thus requires different algorithms. Whereas the parameters that define decision strategies require very fast adaptation, statistical dependencies between inputs reflect invariances in the environment, which could be learned by separate learning processes on much longer time scales.

This article is organized as follows: We present the Bayesian Hebb rule for reinforcement learning tasks in section 3.2, and analyze its convergence behavior for learning reward log-odds. In section 3.3 we present a linear approximation to the Bayesian Hebb rule that is much simpler to implement, but exhibits similar convergence behavior. In section 3.4 we show that after a suitable preprocessing of sensory variables $\mathbf{x}$ one arrives at a population code $\mathbf{y}$ for which optimal decisions can be represented by WTA applied to weighted sums of the variables y_i. The required weights can be learnt quite fast with the Bayesian Hebb rule, even if there exist conditional dependencies among the input variables $\mathbf{x}$. Section 3.5 gives experimental results on the performance of the Bayesian Hebb rule in various action selection tasks. Section 3.5.2 addresses the case of non-stationary reward distributions. In section 3.6 the learning rule is generalized to handle tasks in environments with continuous input signals $\mathbf{x}$. We discuss in section 3.7 salient aspects of the presented results, an application of the Bayesian Hebb rule to model the experimental data of [232], related work, and open problems.

3.2 The Bayesian Hebb rule

In this section we introduce a simple local learning rule, the reward-modulated Bayesian Hebb rule, which learns log-odds of reward probabilities conditioned on binary input variables. Analyzing the convergence behavior of the rule one sees that the true reward log-odds are fixed point attractors for expected weight changes under the reward-modulated Bayesian Hebb rule. The Bayesian Hebb rule also learns fast, since the online learned weights are close to what an optimal Bayesian learning approach, using (biologically unrealistic) counters and auxiliary variables, would achieve. It is further shown that an even simpler rule - which approximates the Bayesian Hebb rule - learns weights which are close to the optimum, and is sufficient for reliable decision making.

3.2.1 Action selection strategies and goals for learning

We consider the standard operant conditioning scenario, where the learner receives at each trial an input $\mathbf{x} = \langle x_1, \ldots, x_m \rangle$ (e.g. a sensory stimulus or internal state signals of the organism) with multinomial variables x_j, chooses an action a out of a set of l possible actions $A = \{a_1, \ldots, a_l\}$, and receives a reward $r \in \{0, 1\}$ with probability $p(r|\mathbf{x}, a)$. The learner's goal is to learn (as fast as possible) a policy $\pi(\mathbf{x}, a) = p(a|\mathbf{x})$ (or $\pi(\mathbf{x})$ in the case of a deterministic policy) so that action selection according to this policy maximizes the average reward. A structural difference to supervised prediction problems is that it does not suffice that the learner passively observes the outcomes of trials, since the reward received for action a in response to stimulus $\mathbf{x}$ provides no information about the probability of rewards for alternative actions a' in response to the same stimulus $\mathbf{x}$. He therefore needs to try out different actions for the same input through an exploration process, in order to learn the reward-probabilities for all actions.

In this article the goal of the learner is fast learning of a policy that approximates the optimal policy. The learner does not necessarily maximize the online performance during learning, and does not specifically try to reduce uncertainty about the outcome of unexplored action. The strategies employed during learning are therefore not Bayes-optimal in the sense of decision theory and sequential analysis [39]. Optimal solutions to the exploration problem for a restricted subclass of tasks can be computed [10, 78, 132], but neural network implementations of these mechanisms are beyond the scope of this article. During learning we follow heuristic strategies that are commonly used

in reinforcement learning [217]. The actions are chosen based on the currently learned weights, which approximate the Bayes optimal estimates for the reward log-odds. In order to maintain a rather high level of rewards during exploration, the agent might for example choose actions stochastically with $p(a|\mathbf{x}) = p(r{=}1|\mathbf{x}, a)$. This corresponds to the *matching behavior* phenomenon observed in biology, where the fraction of choices for one action exactly matches the fraction of total rewards from that action [213]. This policy was used during training in all our computer experiments.

If the goal of the agent is to accumulate as many rewards as possible, and rewards are binary, the agent will choose the action with the highest probability $p(r = 1|\mathbf{x}, a)$ to yield reward. Since the function which maps a probability p onto $\log \frac{p}{1-p}$ is strictly monotonically increasing, the agent can choose instead the action a which has the highest log-odd

$$\log \frac{p(r = 1|\mathbf{x}, a)}{p(r = 0|\mathbf{x}, a)}. \tag{3.2}$$

Hence the optimal policy for maximizing the probability of reward can be written in the form

$$\pi(\mathbf{x}) = \arg\max_{a \in A} \log \frac{p(r = 1|\mathbf{x}, a)}{p(r = 0|\mathbf{x}, a)}. \tag{3.3}$$

We assume for now that the input $\mathbf{x} = \langle x_1, \ldots, x_m \rangle$ consists of m input variables which are arbitrary multinomial discrete random variables with unknown joint distribution (in section 3.6 we will consider the case of continuous inputs $\mathbf{x}$). We assume that these m variables are represented through binary states (firing / non-firing) $\mathbf{y} = \langle y_1, \ldots, y_n \rangle$ of n neurons in a population coding manner. We will define the encoding scheme later in section 3.4 and show that different encodings allow different representations of statistical dependencies. For every possible action a there exists in our simple model (see Figure 3.1) a linear neuron which receives as inputs the components $y_1, \ldots, y_n$ of $\mathbf{y}$. The activation L_a of this linear neuron is defined by the weighted sum

$$L_a = w_{a,0} + \sum_{i=1}^{n} w_{a,i}\, y_i. \tag{3.4}$$

Our approach aims at learning weights $w_{a,i}$ for every action a such that L_a corresponds to the reward log-odd (3.2), which indicates how desirable it is to execute action a in the current situation defined by $\mathbf{x}$ and its neural encoding $\mathbf{y}$. The action with the highest

assumed probability of yielding reward is then selected by a Winner-Take-All (WTA) operation that is formally defined through the binary outputs $z_1, \ldots, z_l$ as follows:

$$z_a = \begin{cases} 1, & \text{if } L_a \geq L_b \text{ for } b \neq a \\ 0, & \text{else} \end{cases} . \tag{3.5}$$

This action selection strategy is commonly referred to as the *greedy* strategy.

If the goal is not only to exploit preceding experience in order to choose an action that maximizes the probability of reward for the current stimulus $\mathbf{x}$, but to simultaneously keep on learning and exploring reward probabilities for other actions, the previously mentioned matching behavior strategy [214] offers an attractive compromise. It can be implemented with the help of the learned parameters $w_{a,i}$ in the following way: The linear gate L_a in Figure 3.1 is replaced by a sigmoidal gate (i.e., the weighted sum L_a according to (3.4) is replaced by $\sigma(L_a) = \frac{1}{1+\exp(-L_a)}$, and the deterministic WTA gate is replaced by a stochastic soft-WTA gate (which selects a as winner with probability $\frac{\sigma(L_a)}{\sum_b \sigma(L_b)}$).

3.2.2 A local rule for learning reward log-odds

We will now present a learning rule and an appropriate input encoding for learning weights, which asymptotically approach target values such that the architecture in Figure 3.1 selects actions optimally. Consider first the case where for a single binary input y_i and action a the reward log-odd $\log \frac{p(r=1|y_i=1,a)}{p(r=0|y_i=1,a)}$ should be learned in the weight $w_{a,i}$. A traditional frequentist's approach would use counter variables

$$\alpha_{a,i} = \#[r = 1 \wedge y_i = 1 \wedge \text{action } a \text{ selected}],$$
$$\beta_{a,i} = \#[r = 0 \wedge y_i = 1 \wedge \text{action } a \text{ selected}]$$

to estimate the reward log-odds $w^*_{a,i}$ after finitely many steps by

$$\hat{w}_{a,i} = \log \frac{\alpha_{a,i}}{\beta_{a,i}} \quad \text{for } i = 1, \ldots, n.$$

In a rewarded trial (i.e. $r = 1$) where $y_i = 1$ and action a is selected this leads to the update

$$\hat{w}^{new}_{a,i} = \log \frac{\alpha_{a,i} + 1}{\beta_{a,i}} = \log \frac{\alpha_{a,i}}{\beta_{a,i}} \left(1 + \frac{1}{\alpha_{a,i}}\right) = \hat{w}_{a,i} + \log(1 + \frac{1}{N_{a,i}}(1 + e^{-\hat{w}_{a,i}})) , \tag{3.6}$$

where $N_{a,i} := \alpha_{a,i} + \beta_{a,i}$ is the total number of previous updates, thus $\frac{1}{\alpha_{a,i}} = \frac{1}{N_{a,i}}(1 + \frac{\beta_{a,i}}{\alpha_{a,i}})$.

Analogously, an update after a new unrewarded trial ($r = 0$) gives rise to the update

$$\hat{w}_{a,i}^{new} = \hat{w}_{a,i} - \log(1 + \frac{1}{N_{a,i}}(1 + e^{\hat{w}_{a,i}})). \tag{3.7}$$

Using the approximation $\log(1 + x) \approx x$, and using a constant learning rate η instead of the factor $\frac{1}{N_{a,i}}$, the update rules (3.6) and (3.7) can be combined to yield a new local learning rule, which does not use any counters. [1] We call this rule the *reward-modulated Bayesian Hebb rule*. The update for weight $w_{a,i}$, whenever action a is selected and $y_i = 1$ is:

$$\boxed{\Delta w_{a,i} = \begin{cases} \eta \cdot (1 + e^{-w_{a,i}}), & \text{if } r = 1 \\ -\eta \cdot (1 + e^{w_{a,i}}), & \text{if } r = 0. \end{cases}} \tag{3.8}$$

This rule increases the weight whenever reward is encountered, and decreases the strength of the synapse otherwise. This learning rule (3.8) is purely local, i.e. it depends only on quantities that are available at the trained synapse, but not on the activity of other presynaptic neurons.

The approximation of the reward-modulated Bayesian Hebb rule to the exact counting model, which computes for every parameter the Bayes-optimal estimate that can be inferred from a fixed finite set of data, is illustrated in Figure 3.2A. In order to estimate a single parameter $q_{a,i} = p(r = 1|y_i = 1, a)$, a uniform prior on $[0, 1]$ was initially imposed on $q_{a,i}$. The counters $\alpha_{a,i}$ and $\beta_{a,i}$, as defined above, were incremented as training samples became available, and the posterior distribution for $q_{a,i}$ was given by the Beta$(\alpha_{a,i} + 1, \beta_{a,i} + 1)$ distribution [156]. The same samples were simultaneously used to update the weight $w_{a,i}$ by rule (3.8). The weights $w_{a,i}$, which represent log-odds $\log \frac{p(r=1|y_i=1,a)}{p(r=0|y_i=1,a)}$ were transformed into probabilities via the transformation

$$\hat{q}_{a,i} = \frac{1}{1 + \exp(-w_{a,i})} \quad .$$

Figure 3.2A shows the optimal posterior for a single $q_{a,i}$ after every update, and the approximation obtained by (3.8). The probability estimated by the Bayesian Hebb rule is always close to the Bayes-optimal estimate.

[1] Using the approximation $\log(1 + x) \approx x$ did not visibly affect the performance of the learning rule in the computer simulations in Section 3.5.

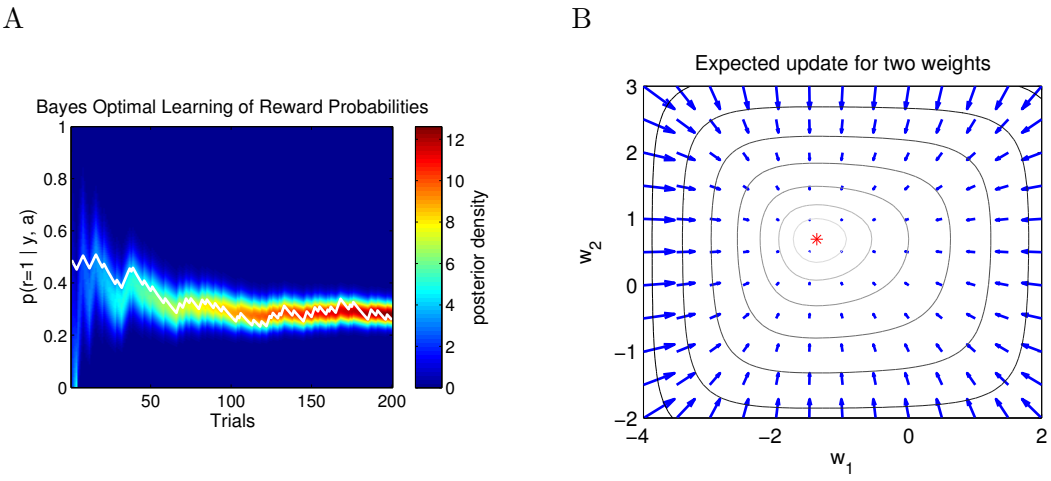

Figure 3.2: Convergence behavior of the Bayesian Hebb rule. **A)** The weights learned by the Bayesian Hebb rule approximate Bayes-optimal learning. The posterior for the reward probability $q_{a,i} = p(r = 1|y_i = 1, a)$ at every training trial was modeled by a Beta($\alpha_{a,i}$+1, $\beta_{a,i}$+1) distribution, with counters $\alpha_{a,i}$ and $\beta_{a,i}$ for rewarded and unrewarded trials. The color shows the estimated posterior density function for $q_{a,i}$ at every training trial. The white curve shows the approximation learned by the Bayesian Hebb rule (3.8) (with constant learning rate $\eta = 0.02$). The weight $w_{a,i}$ was transformed into an estimated reward probability by $\hat{q}_{a,i} = \frac{1}{1+\exp(-w_{a,i})}$. One can see that the approximation follows the optimal estimate closely. **B)** Attractor property of the Bayesian Hebb rule (3.8) plotted for two weights w_1 and w_2. The expected update (indicated by a blue arrow) is always in the direction of the optimal weights (marked by a red star). Gray curves connect points with the same amount of expected weight change.

3.2.3 Convergence properties of the Bayesian Hebb rule in reinforcement learning

The Bayesian Hebb rule is an online learning rule which has no prior knowledge of its target values. However, one can prove that the weights learned with (3.8) converge (in expectation) to their optimal values $w^*_{a,i} = \log \frac{p(r=1|y_i=1,a)}{p(r=0|y_i=1,a)}$, just on the basis of the statistics of pre- and postsynaptic values they encounter. This is in fact very easy to prove, since the equilibrium of the rule is reached when the expected update $E[\Delta w_{a,i}]$ under the rule (3.8) vanishes, and this can be written as

$$E[\Delta w_{a,i}] = 0 \quad \Leftrightarrow \quad p(r = 1|y_i = 1, a) \cdot \eta \cdot (1 + e^{-w_{a,i}}) - \\ -p(r = 0|y_i = 1, a) \cdot \eta \cdot (1 + e^{w_{a,i}}) = 0 \quad .$$

As we show in Appendix 3.8.1, the latter explicitly holds iff $w_{a,i}$ is at the target value $w^*_{a,i} = \log \frac{p(r=1|y_i=1,a)}{p(r=0|y_i=1,a)}$. If a vector of $n+1$ weights $\langle w_{a,0}, \ldots, w_{a,n} \rangle$ for an action a is learned simultaneously, the point $\langle w^*_{a,0}, \ldots, w^*_{a,n} \rangle$ is a global fixed point attractor in the weight space $\mathbb{R}^{n+1}$ with regard to expected weight changes under the Bayesian Hebb rule (see Figure 3.2B).

Another unusual feature of the Bayesian Hebb rule is that one can prove (see Appendix 3.8.1) that it converges exponentially fast to $w^*_{a,i}$ (w.r.t. $E[\Delta w_{a,i}]$). In particular, weight updates move the weight in larger steps towards the attractor $w^*_{a,i}$ if they are farther off, without requiring any change of the learning rate, or knowledge of the ideal values $w^*_{a,i}$.

3.3 The Linear Bayesian Hebb rule

The reward-modulated Bayesian Hebb rule (3.8) includes exponential terms $\exp(-w_{a,i})$ and $\exp(w_{a,i})$. One may argue that an exact calculation of the exponential function is beyond the capabilities of a synaptic learning process. Therefore we have also analyzed a linear approximation to the Bayesian Hebb rule. The exponential function is defined by the Taylor series

$$\exp(x) = \sum_{i=0}^{\infty} \frac{x^i}{i!}. \tag{3.9}$$

Thus, the first order approximations for $\exp(w_{a,i})$ and $\exp(-w_{a,i})$ are

$$\exp(w) \approx 1 + w \tag{3.10}$$

$$\exp(-w) \approx 1 - w. \tag{3.11}$$

Inserting the approximations (3.10) and (3.11) into (3.8), a computationally simpler learning rule is obtained, which we call the *linear Bayesian Hebb* rule. Whenever action a is selected and $y_i = 1$, it updates weight $w_{a,i}$ by:

$$\boxed{\Delta w_{a,i} = \begin{cases} \eta \cdot (2 - w_{a,i}), & \text{if } r = 1 \\ -\eta \cdot (2 + w_{a,i}), & \text{if } r = 0. \end{cases}} \tag{3.12}$$

This new rule resembles strongly the typical Hebb rule with a regularization term. The weights are increased by a constant if the pre- and postsynaptic neurons "fire together" (i.e., $y_i = 1$ and action a is selected), and decreased by a constant if they don't. The

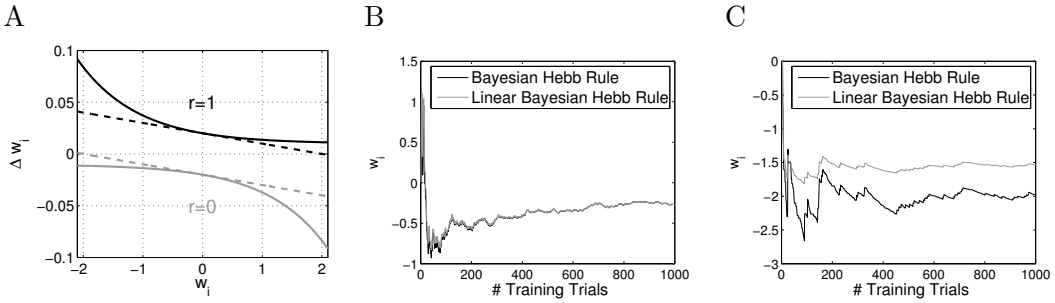

Figure 3.3: Linear approximation of the Bayesian Hebb rule. **A)** Update Δw_i of the Bayesian Hebb rule (3.8) (solid lines) and the linear Bayesian Hebb rule (3.12) (dashed lines) plotted as a function of the current weight value w_i for training trials with $r = 1$ (black curves) and $r = 0$ (gray curves). **B)** Example of the evolution of a single weight under the Bayesian Hebb rule (3.8) and the linear Bayesian Hebb rule (3.12). The target value is close to 0, where the approximation of the linear Bayesian Hebb rule is very good. **C)** Another example of the weight evolution, in which the two rules converge to different weights. The target weight is close to -2, which is the border of the weight-range that the linear Bayesian Hebb rule can cover. The approximation error is therefore large compared to B.

$\pm w_{a,i}$ term prevents the weights from growing too large or too small. Actually, for $\eta \leq 1$ it always keeps the weights within the range $[-2, 2]$. This shows immediately that the linear Bayesian Hebb rule cannot learn the true reward log-odds for arbitrary distributions, but only an approximation. Figure 3.3A shows the updates by the linear Bayesian Hebb rule (dashed lines) in comparison to those of the exact rule (3.8) (solid lines). One can see that the difference between the updates grows for larger values of the target weight $w^*_{a,i}$. However, our computer experiments in Section 3.5 will demonstrate that the linear Bayesian Hebb rule performs remarkably well for many benchmark tasks.

3.3.1 Convergence of the Linear Bayesian Hebb Rule

We show in Appendix 3.8.2 that the equilibrium value for the linear Bayesian Hebb rule (3.12), i.e. the weight value where $E[\Delta w_{a,i}] = 0$, is at

$$\begin{aligned} w^+_{a,i} &= -2 + 4 \cdot p(r = 1 | y_i = 1, a) \\ &= 2 \cdot (p(r = 1 | y_i = 1, a) - p(r = 0 | y_i = 1, a)) \quad . \end{aligned}$$

This equilibrium value is monotonically increasing with $w^*_{a,i}$, the equilibrium value of the exact Bayesian Hebb rule (3.8). They are only equal when $p(r = 1|y_i = 1, a) = p(r = 0|y_i = 1, a)$, i.e. $w^*_{a,i} = w^+_{a,i} = 0$.

In Figures 3.3B and C the evolution of two weights during learning for a random distribution is shown. In 3.3B, the target value is close to zero, where the target values for the exact rule (3.8) and the linear Bayesian Hebb rule (3.12) are very similar. Thus, no big difference in weight space is visible. In 3.3C, however, the target value is close to the maximum value that the linear rule can represent, therefore the two rules do not converge to the same value, indicating a larger approximation error for the linear rule. Hence the linear Bayesian Hebb rule can be expected to perform well if the target values of the weights have small absolute values.

3.4 Population codes for Hebbian learning of asymptotically optimal decisions

In this section two preprocessing mechanisms are presented, which are based on different assumptions about statistical dependencies among input variables. Applied to these population encodings of the input, the WTA circuit in Figure 3.1 selects actions that maximize the probability of obtaining reward, according to the current statistical model represented by the input encoding and the reward log-odds learned with the Bayesian Hebb rule.

We have previously shown that the reward-modulated Bayesian Hebb rule (3.8) has a unique equilibrium at the reward log-odd

$$w^*_{a,i} = \log \frac{p(r = 1|y_i = 1, a)}{p(r = 0|y_i = 1, a)} \quad . \tag{3.13}$$

In order to approximate the true reward probabilities for every action as weighted sums as in (3.4), every vector of input variables $\mathbf{x} = \langle x_1, \ldots, x_m \rangle$ needs to be suitably preprocessed into a population code vector $\mathbf{y} = \langle y_1, \ldots, y_n \rangle$. If the weights $w_{a,i}$ for every y_i and every action a are learned with the Bayesian Hebb rule, our previous analysis guarantees that the resulting policy will asymptotically approach the best policy that can be inferred for the given preprocessing function.

Let the input variables $x_1, \ldots, x_m$ be some arbitrary multinomial random variables with unknown joint distribution, where each variable x_k assumes m_k different values $v_1^k, \ldots, v_{m_k}^k$. For the sake of simplicity we assume that $v_j^k = j$ for $j = 1, \ldots, m_k$ and $k = 1, \ldots, m$.

We first present a very simple population coding, which is sufficient to represent the optimal policy as a weighted sum if the Naive Bayes assumption holds for the input variables, i.e. the input variables x_k are conditionally independent of each other given the selected action a and the reward r:

$$p(x_k|r, a, x_1, \ldots, x_{k-1}, x_{k+1}, \ldots, x_m) = p(x_k|r, a) \text{ for all } k \in \{1, \ldots, m\}. \tag{3.14}$$

In this case it holds that

$$\frac{p(r=1|\mathbf{x}, a)}{p(r=0|\mathbf{x}, a)} = \frac{p(r=1|a)}{p(r=0|a)} \prod_{k=1}^{m} \frac{p(x_k|r=1, a)}{p(x_k|r=0, a)} \,. \tag{3.15}$$

Every x_k is discrete and can only take on finitely many different values. Applying Bayes' theorem and using an indicator function I, which is defined as $I(\texttt{true}) = 1$ and $I(\texttt{false}) = 0$, one can rewrite (3.15) as (see Appendix 3.8.3 for the full derivation)

$$\frac{p(r=1|\mathbf{x}, a)}{p(r=0|\mathbf{x}, a)} = \frac{p(r=1|a)}{p(r=0|a)} \prod_{k=1}^{m} \left(\frac{p(r=0|a)}{p(r=1|a)} \prod_{j=1}^{m_k} \left(\frac{p(r=1|x_k=j, a)}{p(r=0|x_k=j, a)} \right)^{I(x_k=j)} \right). \tag{3.16}$$

This suggests to represent every x_k by a population code, which has $m_k + 1$ binary variables, one for every possible value of x_k, and one bias variable to account for the term $\frac{p(r=0|a)}{p(r=1|a)}$. Formally we define the simple preprocessing (*SP*) $\boldsymbol{\phi}(x_k)$ for a single variable x_k as

$$\boldsymbol{\phi}(x_k) = [-1, \varphi_1, \ldots, \varphi_{m_k}]^T, \text{ where } \varphi_j = \begin{cases} 1, & \text{if } x_k = j \\ 0, & \text{otherwise.} \end{cases} \tag{3.17}$$

As an example we consider the simple reward distribution with 2 input variables $\mathbf{x} = \langle x_1, x_2 \rangle$, modeled by the Bayesian network in Figure 3.4A. Under the Naive Bayes assumption the dependency of x_2 on the input variable x_1 is neglected, i.e. the arrow $x_1 \to x_2$ in the Bayesian network is ignored. For binary x_k, the population code under this assumption is illustrated in Figure 3.4C. Each input variable x_k is encoded separately by 3 variables y_i, where one is constantly -1, and only one other y_i is active, depending on the value of x_k.

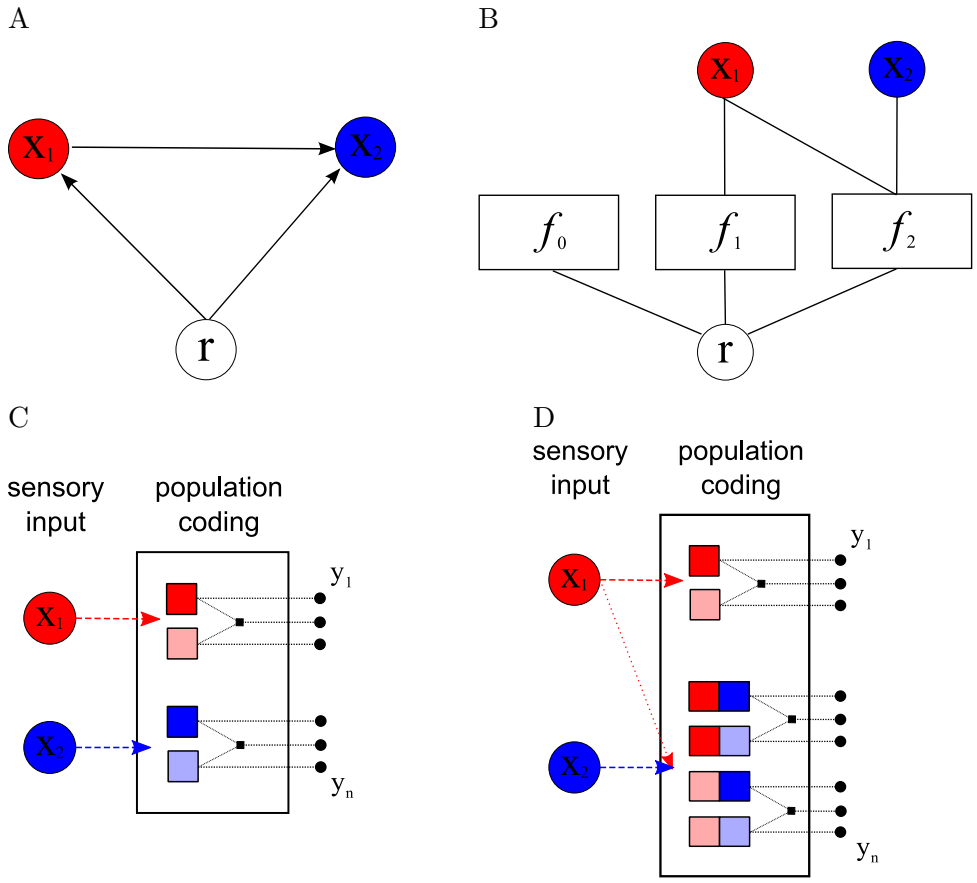

Figure 3.4: Preprocessing for tasks with arbitrary statistical dependencies. **A)** An example Bayesian network for the joint distribution of sensory inputs $\mathbf{x} = \langle x_1, x_2 \rangle$ and reward r. **B)** Factor graph representation for the prediction of r, according to the Bayesian network in panel A. Here, f_0 represents the prior $p(r)$, and the factors f_1 and f_2 represent the conditional probabilities $p(x_1|r)$ and $p(x_2|x_1, r)$, respectively. **C)** Population coding under the Naive Bayes assumption, which we refer to as simple preprocessing (SP). For every possible value of the variables x_k (here x_1, x_2 are binary), there is one variable y_i (indicated by a black circle) that outputs the value 1. Additionally there is one variable y_i for every x_k, which is constantly at -1 (black square). The constant bias term y_0 is not shown. **D)** Population coding applied to the factors in the factor graph shown in panel B. For each combination of values of the variables $\{x_k, \mathbf{x}_{\mathrm{P}_k}\}$ of a factor there is exactly one variable y_i (indicated by a black circle) associated with the factor that outputs the value 1. Other variables y_i represent OR's of these values (black squares), and yield either 0 or -1. The constant bias term y_0 is not shown. We refer to the resulting preprocessing circuit that maps sensory inputs $\mathbf{x}$ onto internal variables $\mathbf{y}$ that support Hebbian learning of optimal decisions as generalized preprocessing (GP).

The vectors $\boldsymbol{\phi}(x_k)$ for $k = 1, \ldots, m$ are concatenated into one population code vector $\mathbf{y}$ for the whole input. $\mathbf{y}$ has $n = 1 + m + \sum_{k=1}^{m} m_k$ entries, of which exactly $2 \cdot m + 1$ are non-zero, and the first entry $y_0 \equiv 1$ corresponds to the bias term $\frac{p(r=1|a)}{p(r=0|a)}$ in (3.16):

$$\mathbf{y} = \boldsymbol{\Phi}(\mathbf{x}) = \begin{bmatrix} 1 \\ \boldsymbol{\phi}(x_1) \\ \boldsymbol{\phi}(x_2) \\ \vdots \\ \boldsymbol{\phi}(x_m) \end{bmatrix} . \tag{3.18}$$

Substituting the definition of $\mathbf{y}$ from (3.17) and (3.18) into (3.16) and taking the logarithm then yields the log-odd function

$$\log \frac{p(r=1|\mathbf{y},a)}{p(r=0|\mathbf{y},a)} = \log \frac{p(r=1|a)}{p(r=0|a)} + \sum_{i=1}^{n} y_i \log \frac{p(r=1|y_i \neq 0,a)}{p(r=0|y_i \neq 0,a)} . \tag{3.19}$$

If we use the population code (3.18) for $\mathbf{y}$, we can apply the reward-modulated Bayesian Hebb rule (3.8) for every y_i to learn reward log-odds conditioned on feature y_i being active[1]. For a y_i that is constantly active, such as y_0, the weight $w_{a,i}$ will converge to the prior reward probability $\log \frac{p(r=1|a)}{p(r=0|a)}$ for action a. Inserting the target values (3.13) of the weights into (3.19), we can therefore write

$$\log \frac{p(r=1|\mathbf{y},a)}{p(r=0|\mathbf{y},a)} = \sum_{i=0}^{n} w^*_{a,i}\, y_i . \tag{3.20}$$

During learning the current values of the weights $w_{a,0}, \ldots, w_{a,n}$ are used to approximate the true reward log-odd for every action a as the weighted sums in (3.4). Actions are selected by a heuristic method according to their predicted probability of yielding reward (e.g. greedy or matching behavior). If the Naive Bayes assumption holds, the reward-modulated Bayesian Hebb rule in combination with a simple population coding for every input variable x_k is therefore sufficient to asymptotically learn the optimal action selection policy.

3.4.1 Learning decisions for arbitrary discrete distributions

We now address the more general case, where conditional independence of the input variables $x_1, \ldots, x_m$ cannot be assumed. We show that with a fixed preprocessing of

[1] We consider a feature y_i active if it is non-zero, i.e. both $y_i = 1$ and $y_i = -1$ are *active* features.

the input that takes their dependencies into account, the Bayesian Hebb rule enables the resulting neural network to converge quite fast to the best performance that any action selection mechanism could possibly achieve. The dependency structure of the underlying input and reward distribution is given in terms of an arbitrary Bayesian Network BN for discrete variables (like e.g. Figure 3.4A). BN can be represented, like every Bayesian network, by a directed graph without directed cycles. We do not assume any further restrictions on the structure of the Bayesian network, so BN does not have to be a tree (as assumed in [45]), and it is not required to have no undirected cycles (as necessary for guaranteed convergence of belief propagation algorithms [22]).

Without loss of generality we choose a numbering scheme such that the direct children of the reward node r in BN are $x_1, \ldots, x_{m'}$. The dependencies in BN can be described by $m+1$ parent sets $\mathtt{P}_k$, which are possibly empty, and explicitly exclude the reward node r. $\mathtt{P}_k$ is thus defined as

$$\mathtt{P}_k \;=\; \{i \mid \text{a directed edge } x_i \to x_k \text{ exists in BN} \;\text{ and }\; x_i \neq r\} \quad .$$

Additionally we define $\mathtt{P}_r$ as the set of all parents of the reward-node r. The joint probability distribution on the variables $r, x_1, \ldots, x_m$ in the Bayesian network for action a can then be factored, giving rise to a factor graph [130] as indicated in Figure 3.4B:

$$p(r, \mathbf{x}|a) = p(r|\mathbf{x}_{\mathtt{P}_r}, a) \prod_{k=1}^{m'} p(x_k|\mathbf{x}_{\mathtt{P}_k}, r, a) \prod_{k=m'+1}^{m} p(x_k|\mathbf{x}_{\mathtt{P}_k}, a). \tag{3.21}$$

When calculating the log-odd of obtaining reward or not, the last terms in (3.21) cancel out, and a simple application of Bayes' theorem leads to

$$\begin{aligned}\log \frac{p(r=1|\mathbf{x}, a)}{p(r=0|\mathbf{x}, a)} = &\log \frac{p(r=1|\mathbf{x}_{\mathtt{P}_r}, a)}{p(r=0|\mathbf{x}_{\mathtt{P}_r}, a)} + \\ &+ \sum_{k=1}^{m'} \left(\log \frac{p(r=1|x_k, \mathbf{x}_{\mathtt{P}_k}, a)}{p(r=0|x_k, \mathbf{x}_{\mathtt{P}_k}, a)} - \log \frac{p(r=1|\mathbf{x}_{\mathtt{P}_k}, a)}{p(r=0|\mathbf{x}_{\mathtt{P}_k}, a)} \right). \end{aligned} \tag{3.22}$$

This is a sum of conditional reward log-odds, which can all be learned with the reward-modulated Bayesian Hebb rule. We now develop a suitable sparse encoding of $x_1, \ldots, x_m$ into binary variables $y_1, \ldots, y_n$ (with $n \gg m$), such that the reward log-odd can be written as a weighted sum

$$\log \frac{p(r=1|\mathbf{y}, a)}{p(r=0|\mathbf{y}, a)} = \sum_{i=1}^{n} w_{a,i}\; y_i,$$

and the weights $w_{a,i}$ correspond to conditional reward log-odds of y_i's. For the example Bayesian network in Figure 3.4A, the corresponding sparse code is illustrated in Figure 3.4D: One binary variable is created for every possible value assignment to a variable x_k and all its parents $\mathbf{x}_{\mathtt{P}_k}$, and additional binary variables are created for every possible value assignments to the parent nodes only. One should contrast this with the simple population code in Figure 3.4C, which assumes that the Naive Bayes condition holds, and therefore ignores that x_2 is dependent on x_1.

BN can also be viewed as a factor graph (see Figure 3.4B), in which there is for every variable x_k a factor f_k, which is connected to r, x_k and $\mathbf{x}_{\mathtt{P}_k}$, the parents of x_k in BN. The preprocessing is then computed separately for every factor f_k. We define the fixed *generalized preprocessing* (*GP*) operation for f_k with $k \geq 1$ as

$$\mathbf{\Phi}(x_k, \mathbf{x}_{\mathtt{P}_k}) = \begin{bmatrix} \boldsymbol{\phi}(x_k, \mathbf{x}_{\mathtt{P}_k}) \\ -\boldsymbol{\phi}(\mathbf{x}_{\mathtt{P}_k}) \end{bmatrix}. \tag{3.23}$$

The summands of the sum on the r.h.s. of (3.22) are split into two parts, and $\boldsymbol{\phi}(x_k, \mathbf{x}_{\mathtt{P}_k})$ defines the preprocessing for the first part, whereas $-\boldsymbol{\phi}(\mathbf{x}_{\mathtt{P}_k})$ defines the preprocessing for the latter part. The variables $\langle x_k, \mathbf{x}_{\mathtt{P}_k} \rangle$ are viewed as a single multinomial variable, and $\boldsymbol{\phi}(x_k, \mathbf{x}_{\mathtt{P}_k})$ is a representation of this multinomial variable through simple population coding. Thus, $\boldsymbol{\phi}(x_k, \mathbf{x}_{\mathtt{P}_k})$ has as many binary output variables $y_{k,i}$ as there are different assignments of values to all variables in $\langle x_k, \mathbf{x}_{\mathtt{P}_k} \rangle$, and exactly one variable $y_{k,i}$ has value 1 for each such assignment. Let $y_{k,i}$ be the binary output variable that corresponds to some assignment $x_k = j,\ \mathbf{x}_{\mathtt{P}_k} = \mathbf{u}$, then the corresponding weight $w_{a,k,i}$ for action a can be learnt through the same reward-modulated Bayesian Hebb rule (3.8) as in the Naive Bayes case. The target value, to which $w_{a,k,i}$ will converge is then

$$w^*_{a,k,i} = \log \frac{p(r=1|y_{k,i}=1,a)}{p(r=0|y_{k,i}=1,a)} = \log \frac{p(r=1|x_k=j,\mathbf{x}_{\mathtt{P}_k}=\mathbf{u},a)}{p(r=0|x_k=j,\mathbf{x}_{\mathtt{P}_k}=\mathbf{u},a)} \quad . \tag{3.24}$$

Analogously, the application of the reward-modulated Bayesian Hebb rule (3.8) for every component $y_{\mathtt{P}_k,i}$ of $-\boldsymbol{\phi}(\mathbf{x}_{\mathtt{P}_k})$ leads to the target weights

$$w^*_{a,\mathtt{P}_k,i} = \log \frac{p(r=1|y_{\mathtt{P}_k,i}=-1,a)}{p(r=0|y_{\mathtt{P}_k,i}=-1,a)} = \log \frac{p(r=1|\mathbf{x}_{\mathtt{P}_k}=\mathbf{u},a)}{p(r=0|\mathbf{x}_{\mathtt{P}_k}=\mathbf{u},a)} \quad , \tag{3.25}$$

with the only formal modification to the update rule (3.8) being that updates are not only made when $y_i = 1$, but also when $y_i = -1$, which obviously does not change

the behavior of the learning process. Formally, all preprocessed vectors $\mathbf{\Phi}(x_k, \mathbf{x}_{\mathrm{P}_k})$ are concatenated into one vector $\mathbf{y}$ with $n = \sum_{k=1}^{m'} N_k + N_{\mathrm{P}_k}$ entries

$$\mathbf{y} = \begin{bmatrix} \mathbf{\Phi}(\mathbf{x}_{\mathrm{P}_r}) \\ \mathbf{\Phi}(x_1, \mathbf{x}_{\mathrm{P}_1}) \\ \vdots \\ \mathbf{\Phi}(x_{m'}, \mathbf{x}_{\mathrm{P}_{m'}}) \end{bmatrix} .$$

This sparse, redundant input encoding provides a weighted sum representation of the reward log-odd

$$\log \frac{p(r=1|\mathbf{y}, a)}{p(r=0|\mathbf{y}, a)} = \sum_{i=1}^{n} w_{a,i}\ y_i,$$

where the weights $w_{a,1}, \ldots, w_{a,n}$ can all be learnt through the reward-modulated Bayesian Hebb rule (3.8) as described above.

3.5 Results of Computer Simulations

We now evaluate the performance of the reward-modulated Bayesian Hebb rule and its linear approximation and compare it to the standard learning model for simple conditioning tasks, the non-Hebbian Rescorla-Wagner rule [184].

The reward-modulated Bayesian Hebb rule (3.8) was tested on a variety of action selection tasks with 4 possible actions. A Bayesian network with dependency structure as in Figure 3.4A was used to model the distribution $p(r, x_1, x_2|a)$ for every action a, where r is the binary reward signal, and x_1, x_2 are the two binary input signals. We assigned a constant reward prior $p(r|a) = 0.25$ to every action a, and randomly generated the conditional probability tables for $p(x_1|r, a)$ and $p(x_2|x_1, r, a)$: for every action a, every x_k ($k \in \{1, 2\}$), and every possible value assignment to the parent nodes $\langle \mathbf{x}_{\mathrm{P}_k}, r \rangle$, a random sample $q \in [0, 1]$ was drawn from a Beta-distribution, and $p(x_k = 1|\mathbf{x}_{\mathrm{P}_k}, r, a)$ was set to q.

The Bayesian networks which model the reward distribution were also used to create the samples of input vectors $\mathbf{x} = \langle x_1, x_2 \rangle$ for every training trial. First, one of the four Bayesian networks was chosen randomly with equal probability, so the distribution of input or test samples does not depend on the action selection during learning. Inputs $\mathbf{x}$ were drawn as random samples from the selected network. The agent then received the input $\mathbf{x}$ and chose its action a. The binary reward signal r was sampled from

the distribution $p(r|\mathbf{x}, a)$, and thus depends on the chosen action. The agent used the tuple $\langle \mathbf{x}, a, r \rangle$ to update its weights $w_{a,i}$. Training consisted of 2000 trials, in which the matching behavior strategy (see section 3.2.1) was used for action selection during learning. The evaluation of the performance of the resulting policy after every trial used the greedy strategy (3.3), choosing actions on 500 independent test trials and measuring the average reward. The experiment was averaged over 250 different tasks with different reward distributions.

The preprocessed binary vectors $\mathbf{y} = \mathbf{\Phi}(\mathbf{x}) \in \{0, 1\}^n$ were created either by simple population coding (see (3.18) and Figure 3.4C), which is suitable for the Naive Bayes case (3.14), or generalized preprocessing (see (3.23) and Figure 3.4D). The former mechanism is referred to as *Bayesian Hebb SP* in Figure 3.5 and the remainder of this article, whereas the generalized preprocessing mechanism is referred to as *Bayesian Hebb GP*. The Bayesian Hebb rule with these two kinds of preprocessing mechanisms was compared to the non-Hebbian Rescorla-Wagner rule [184]. This rule predicts the value of a (multi-dimensional) stimulus as a linear sum,

$$V(\mathbf{y}) = w_0 + \sum_{i=1}^{n} w_i y_i \quad ,$$

and minimizes the prediction error with a delta learning rule

$$\Delta w_i = \eta y_i \left(r - w_0 - \sum_{i=1}^{n} w_i y_i \right) \quad . \tag{3.26}$$

It can be seen from equation (3.26), that for the update of a single weight, the complete prediction of value for the current state, which depends on all weights, is needed. In the experiments the Rescorla-Wagner rule was used to learn weights for every action separately. The classical Rescorla-Wagner rule (3.26) , which we use for comparison, is directly applied to the inputs $\mathbf{x}$. We show in Appendix 3.8.5 that the performance and learning speed of Rescorla-Wagner can also be improved if it is applied to the preprocessed vectors $\mathbf{y} = \mathbf{\Phi}(\mathbf{x})$, using the same *SP* and *GP* preprocessing mechanisms as for the Bayesian Hebb rule.

In addition, the reward-modulated Bayesian Hebb rule was also compared to a Bayes-optimal weight learning rule. In this case the conditional probabilities in the Bayesian network in Figure 3.4A were estimated using counter variables (see section 3.2.2), and exact inference was used to compute reward probabilities for every action.

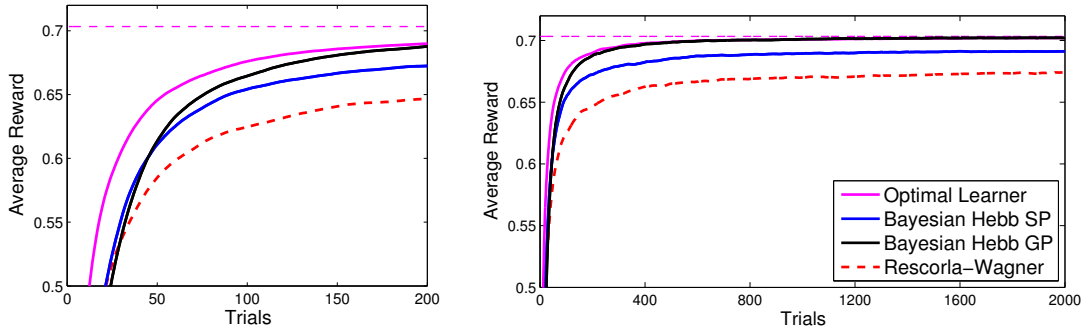

Figure 3.5: Performance of the reward-modulated Bayesian Hebb rule for action selection in a 4-action task with stochastic rewards. Each learner was trained on 2000 trials, and after every trial the performance was measured as the average reward of the greedy policy of each learner on 500 independent test trials (left: performance during the first 200 training trials). The results were averaged over 250 different problems, all having the statistical dependency structures as in Figure 3.4A, but random reward distributions (average learning and preprocessing time per problem on a dual-core 2.66 GHz, 16GB RAM PC: 0.9 s for SP, and 4.1 s for GP). The horizontal dashed line reflects the best possible performance of an optimal policy. The Bayesian Hebb rule with simple population coding (*Bayesian Hebb SP*) and generalized preprocessing (*Bayesian Hebb GP*) were compared to action-learning with the non-Hebbian Rescorla-Wagner rule. The learning rate was set to $1/N_{a,i}$, and stochastic action selection was used for exploration during training. The Bayesian Hebb rule for both preprocessing methods learned faster than the non-Hebbian Rescorla-Wagner rule and converged to better policies. With generalized preprocessing, the Bayesian Hebb rule converged to the optimal action-selection policy, as predicted by the theoretical analysis. Error bars are in the range of 10^{-3} and are omitted for clarity.

Figure 3.5 shows that the reward-modulated Bayesian Hebb rule for both types of preprocessing learns faster than the non-Hebbian Rescorla-Wagner rule and converges to better policies. If generalized preprocessing is used, the learned policy after approximately 200 trials is almost indistinguishable from the policy of an optimal learner, and after approximately 1000 trials the performance is very close to the optimal performance level.

3.5.1 Approximations to the Bayesian Hebb rule

We have shown in section 3.3 that the linear Bayesian Hebb rule (3.12) can be derived as a first-order Taylor approximation of the reward-modulated Bayesian Hebb rule

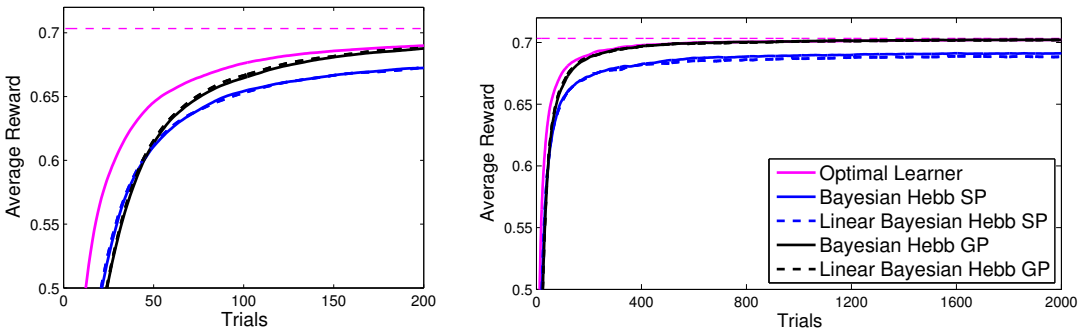

Figure 3.6: Performance of the linear approximations to the reward-modulated Bayesian Hebb rule in the same 4-action tasks as in Figure 3.5 (left: performance during the first 200 training trials). Both for simple population coding (*SP*) and generalized preprocessing (*GP*), the linear approximation to the learning rule learned as well as the exact rule. Error bars are in the range of 10^{-3} and are omitted for clarity.

(3.8). There are no theoretical guarantees that the linear Bayesian Hebb rule will asymptotically converge towards weight values that allow optimal decision making. We compared the two rules on the same random Bayesian network tasks for action selection empirically, using both the simple preprocessing (*SP*) for the Naive Bayes case, and the generalized preprocessing (*GP*) for arbitrary reward distributions. Figure 3.6 shows that this even simpler rule found good policies as quick as the exact rule. The quality of the final policy was almost indistinguishable from the policies found by the exact Bayesian Hebb rule.

3.5.2 Adaptation to changing reward distributions

In most realistic scenarios an organism experiences during its lifetime changes in the environment in which it lives. It is therefore important that a learning rule can adapt quickly to a changing reward or input distribution. It is clear that a learning rate that decays with $\frac{1}{N_i}$ (where N_i is the number of updates for a weight w_i) is not suitable for changing environments. We therefore used for this task the variance tracking mechanism for learning rate adaptation, which was first introduced by [158]. This mechanism keeps track of the variance of each weight, and adapts learning rates accordingly. Learning rates are reduced for weights with small fluctuations, whereas they are increased for weights with high variance, which is an indication that those weights have not yet

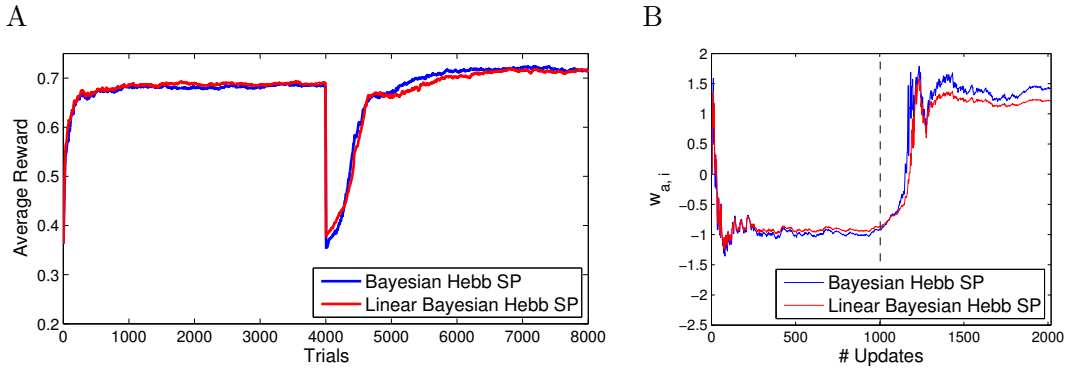

Figure 3.7: Behavior of the Bayesian Hebb rule when the reward distribution changes during training. **A)** Performance of the agent if a new reward distribution is introduced after 4000 training trials. There is an immediate drop when the distribution changes, but good performance is recovered quickly by both rules. **B)** Evolution of a single weight $w_{a,i}$ when the reward distribution changes. The weights are plotted at every trial where action a is selected, and an update for the plotted weight occurs. The weight first settles at the desired value for the first distribution, and then quickly adapts to the new target value when the distribution changes (indicated by the black dashed line).

settled at their equilibrium values.

The learning rate adaptation mechanism uses two auxiliary variables, which can be locally estimated for every weight w_i: a running average of the weight is computed in $\bar{w}_i$, and a running average of the squared weight in $\bar{q}_i$, using the following simple update rules:

$$\begin{aligned} \bar{w}_i^{new} &\leftarrow (1-\eta_i)\,\bar{w}_i + \eta_i\, w_i \\ \bar{q}_i^{new} &\leftarrow (1-\eta_i)\,\bar{q}_i + \eta_i\, w_i^2 \quad . \end{aligned} \tag{3.27}$$

With these values the short-time variance of each weight can be estimated as $\bar{q}_i - \bar{w}_i^2$. Assuming that samples are drawn from stationary input distributions, it was shown in [158] that the variance of a weight w_i can be related to the sample size N_i in the Bayes-optimal learning case (see also section 3.2.2), where exact counters for all combinations of inputs, actions and rewards are used, and conditional reward probabilities are modeled with Beta-distributions. According to this analysis, the new learning rate η_i^{new} can be set as

$$\eta_i^{new} \leftarrow \frac{\bar{q}_i - \bar{w}_i^2}{1+\cosh \bar{w}_i} \quad . \tag{3.28}$$

In practice this mechanism decays like $\frac{1}{N_i}$ under stationary conditions. It can also handle changing input distributions, because a new target value for w_i leads to larger updates Δw_i, thus increasing the short-time variance of the weight, and by (3.28) the learning rate η_i.

The variance tracking mechanism is an analytically justified rule for setting learning rates. Biological implementations of qualitatively similar processes are plausible, since all auxiliary quantities can be observed locally at the synapse. What is required is essentially a process that locally modulates potentiation or depression of synapses, and itself is dependent on the magnitude of recent local synaptic weight changes. This could in principle be achieved by a large variety of metaplasticity mechanisms that are known to modulate synaptic plasticity (see [2] for a recent review). Neuromodulators such as acetylcholine and norepinephrine could play a special role in the control of learning rates and the reduction of oscillations of weight updates [53, 233].

In the experiment shown in Figure 3.7, the weights were learned in 4000 training trials, after which the environment was changed and the learner was trained for another 4000 trials on the new input and reward distributions. Figure 3.7A shows that the performance of the learners initially improved, then dropped as soon as the distributions were switched, but quickly adapted to the new distribution, reaching almost the same performance. Figure 3.7B shows the evolution of a single weight in this scenario, for all trials in which it was updated. It can be seen that the weight first settled around the equilibrium value of the first distribution, and grew to reach the new target value after the switch.

3.5.3 Simulations for large input and action spaces

The Bayesian Hebb rule also works well for significantly larger problems. The same algorithms as in the previous sections were applied to problems with 100 binary input attributes, and 10 possible actions. The structures of the Bayesian networks that define the reward distributions for every action were generated randomly, using the algorithm described in [106]. Every node in the network could have a maximum of 5 parent nodes. The protocol for the generation of training samples and rewards was the same as for the previous experiments (see beginning of section 3.5). During learning actions were selected randomly, and the greedy policy was used for the evaluation on 1000 independent test trials (once every 1000 training trials).

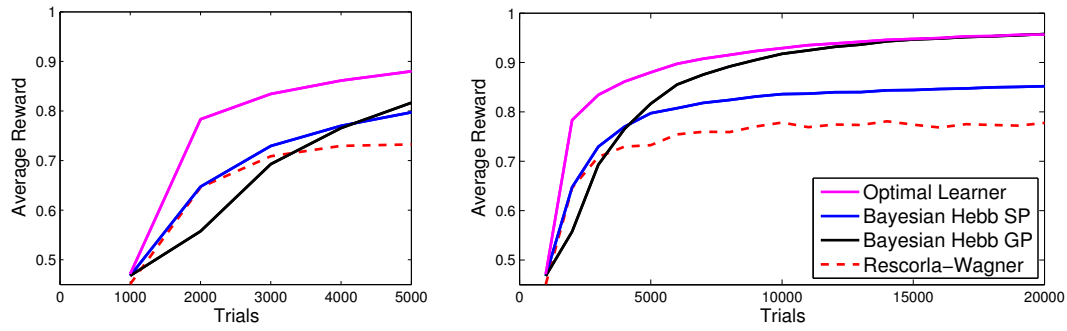

Figure 3.8: The Bayesian Hebb rule works well also for simulations with large input and action spaces. Each learner was trained on 20,000 trials of action selection problems with 10 actions, 100 binary input attributes, and stochastic rewards. Every 1000 trials the performance was measured as the average reward of the greedy policy of each learner on 1000 independent test trials (left: performance during the first 5000 training trials). The results were averaged over 40 different problems with random statistical dependency structures and random reward distributions (average learning and preprocessing time per problem on a 2-core 2.66 GHz, 16GB RAM PC: 27.8 s for SP, and 301.6 s for GP). The learning rates were set to $1/N_{a,i}$, and random action selection was used for exploration during training. With generalized preprocessing, the Bayesian Hebb rule approached the performance of an optimal learning mechanism. Error bars are in the range of 10^{-2} and are omitted for clarity.

Figure 3.8 shows that the Bayesian Hebb rule learns fast, both for simple population coding (*SP*), and generalized preprocessing (*GP*). The latter initially performs worse than SP, because the number of weights to learn is very large (about 1000 weights for every action), and approximation errors sum up. Given more training data, the Bayesian Hebb rule with generalized preprocessing approaches the performance of an optimal learner. The linear approximations to the reward-modulated Bayesian Hebb rule perform equally well on this task for both types of preprocessing.

3.6 Decision making with continuous inputs

The Bayesian Hebb rule can be generalized to action-selection problems defined on continuous input distributions. A rule very similar to (3.8) learns reward log-odds on a continuous input encoding, comparable to population codes with bell-shaped tuning curves that are observed in the brain.

The Bayesian Hebb rule has previously been defined only for discrete inputs x_k, which were mapped to binary variables y_i with various ways of preprocessing. We now present a learning rule to approximate distributions of a binary reward variable for continuous inputs. The preprocessing for this case is a population code, which uses radial-basis functions (*RBFs*) [1] to map continuous input variables x_k to new continuous features y_i, which may e.g. correspond to firing rates in a neural population code. Population codes with RBF- or bell-shaped tuning curves have been observed, for example, in area MT of the visual system for direction sensitive cells (see [179] for a review), place cells in rat hippocampus [165], or for the encoding of movement directions in primate motor cortex [71]. Networks of RBF units are also commonly used for models of visual object recognition [187].

Consider the input variables $\mathbf{x} = \langle x_1, \ldots, x_m \rangle \in X \subseteq \mathbb{R}^m$, and a binary reward variable $r \in \{0, 1\}$. The continuous input $\mathbf{x}$ is mapped to a new set of n continuous non-negative features y_i. The activation of feature y_i is proportional to the activation of a RBF-kernel $\phi_i(\mathbf{x})$:

$$\phi_i(\mathbf{x}) = \exp\left(-\sum_{k=1}^{m} \frac{|x_k - c_{i,k}|^2}{s_{i,k}^2}\right). \tag{3.29}$$

The centers of the RBF kernels are located at $\mathbf{c}_i = \langle c_{i,1}, \ldots, c_{i,m} \rangle$, and the widths of the kernels are given by $s_{i,k}$ (different widths may be used for different input dimensions). The preprocessed vector $\mathbf{y} = \langle y_1, \ldots, y_n \rangle$ is obtained by calculating the activations of all n different RBFs and normalizing the vector:

$$y_i(\mathbf{x}) = \frac{\phi_i(\mathbf{x})}{\sum_{j=1}^{n} \phi_j(\mathbf{x})}. \tag{3.30}$$

Notice that this kind of preprocessing can take combinations of variables into account, such as RBF kernels on $\mathbb{R}^m$, not only single variables. Figure 3.9 illustrates a simple continuous population code for 5 RBF kernels in one input dimension.

A rule for learning reward log-odds conditioned on a single feature $y_i = y_i(\mathbf{x})$ can be defined by generalizing the reward-modulated Bayesian Hebb rule (3.8). Whenever action a is selected, every weight $w_{a,i}$ is updated by:

$$\boxed{\Delta w_{a,i} = \begin{cases} \eta \cdot y_i(\mathbf{x}) \cdot (1 + e^{-w_{a,i}}), & \text{if } r = 1 \\ -\eta \cdot y_i(\mathbf{x}) \cdot (1 + e^{w_{a,i}}), & \text{if } r = 0 \quad . \end{cases}} \tag{3.31}$$

[1] Other mappings are also possible, but are not presented in this paper.

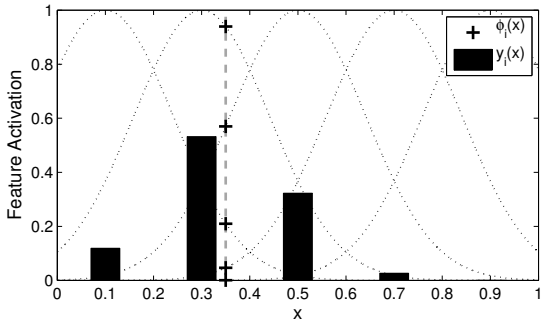

Figure 3.9: Example of a continuous population code with 5 equally spaced RBF kernels (width $s = 0.2$) for a 1-dimensional input x. The activations of the RBF-kernels $\phi_i(x)$ depend on the distance between x and the center c_i of the kernel. The normalized features $y_i(x)$ are obtained by dividing every $\phi_i(x)$ by the total sum of activations. The RBF-kernel activations $\phi_i(x)$ (black crosses mark the intersection of the vertical line at $x = 0.35$ with the 5 RBF-kernels indicated by dotted lines), and the normalized feature activations $y_i(x)$ (dark bars) are here shown for an example input at $x = 0.35$ (gray dashed line).

This rule is a generalization of rule (3.8), in which the updates are weighted by the activation of feature y_i. For the previously described discrete population codes, where y_i is either 0 or 1, the rule (3.31) is equivalent to (3.8).

For the analysis of the equilibrium of rule (3.31), we use an alternative population code of virtual binary features $\tilde{y}_1, \ldots, \tilde{y}_n$. We interpret $y_1(\mathbf{x}), \ldots, y_n(\mathbf{x})$ as (non-normalized) probabilities for randomly selecting one $i \in \{1, \ldots, n\}$, for which one sets $\tilde{y}_i = 1$ (while setting $\tilde{y}_j = 0$ for $j \neq i$). This gives a new interpretation to the continuous population code features $y_i(\mathbf{x})$, because they are proportional to the probability that $\tilde{y}_i = 1$ (we then say that "feature $\tilde{y}_i$ is active").

To find the equilibrium of the rule (3.31) for the weight $w_{a,i}$, we set the expected update $E[\Delta w_{a,i}]$ to zero, and rewrite it as

$$\begin{aligned} E[\Delta w_{a,i}] = 0 \quad \Leftrightarrow \quad & (1 + e^{-w_{a,i}}) \int_X y_i(\mathbf{x})\ p(r = 1, \mathbf{x}|a)\ d\mathbf{x} \\ & -(1 + e^{w_{a,i}}) \int_X y_i(\mathbf{x})\ p(r = 0, \mathbf{x}|a)\ d\mathbf{x} = 0 \quad . \end{aligned}$$

It is shown in Appendix 3.8.4 that this condition is fulfilled if and only if $w_{a,i}$ is at the

target value

$$w^*_{a,i} = \log \frac{p(r=1|\tilde{y}_i=1,a)}{p(r=0|\tilde{y}_i=1,a)} \quad .$$

If the active (virtual) feature $\tilde{y}_i$ was known, the corresponding weight $w_{a,i}$ would directly indicate the log-odd of obtaining reward with action a. In this scenario, however, only the continuous features $y_i(\mathbf{x}), i = 1, \ldots, n$ are known. Due to the normalization, the feature values sum up to 1, and one can therefore weight every $w_{a,i}$ by $y_i(\mathbf{x})$, yielding

$$L_a(\mathbf{x}) = \sum_{i=1}^{n} w_{a,i}\ y_i(\mathbf{x}) \quad , \tag{3.32}$$

which is an interpolation between the reward log-odds $w_{a,i}$ for different features $\tilde{y}_i$. The interpolation weights are in this case the factors $y_i(\mathbf{x})$, which means that those features $\tilde{y}_i$ which are more likely to be active contribute more to the weighted sum, since $y_i(\mathbf{x})$ is proportional to $p(\tilde{y}_i = 1|\mathbf{x})$. $L_a(\mathbf{x})$ thus approximates the reward log-odd $\log \frac{p(r=1|\mathbf{x},a)}{p(r=0|\mathbf{x},a)}$, and the reward probability $p(r = 1|\mathbf{x}, a)$ can be approximated by

$$p(r=1|\mathbf{x},a) \approx \sigma(L_a(\mathbf{x})) = \frac{1}{1+e^{-L_a(\mathbf{x})}} \quad , \tag{3.33}$$

where $\sigma(.)$ is the log-sigmoidal transfer function.

3.6.1 Computer Experiments with continuous input

For the following experiment reward distributions were defined on single continuous input variables $x \in [0, 1]$. For every action a different reward distribution was modeled, and the learner's task was to approximate the true reward distributions with the continuous Bayesian Hebb rule (3.31), and to choose the action with the highest reward probability. 2000 training trials with inputs drawn from a uniform distribution on $[0, 1]$ were used, and the performance after every update was measured on 500 independent test trials. 20 RBFs with constant widths $s = 0.05$ were used for the input preprocessing. The centers of the RBFs were equally distributed in the interval $[0, 1]$. Figure 3.10 shows the performance at every training trial, and the approximations of the reward distributions that were obtained after 2000 training trials. The average reward obtained after training is close to the best possible performance, and the reward distributions are learned accurately.

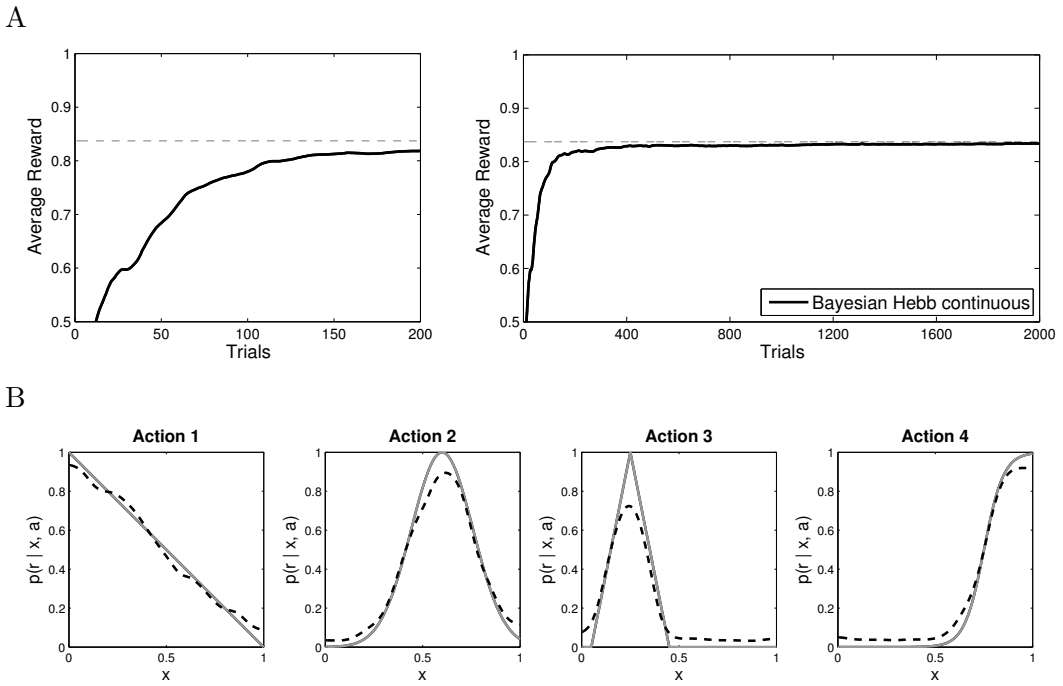

Figure 3.10: Performance of the Bayesian Hebb rule for continuous inputs. The input preprocessing consists of 20 RBF kernels that yield a population code $\mathbf{y}$ for the continuous inputs $\mathbf{x}$. **A)** Average reward of the learner obtained on 500 independent test trials during training on 2000 trials (left: performance during the first 200 training trials). The performance level rises quickly and in the end is close to the best possible performance of an optimal action selector (horizontal dashed line). Error bars are in the range of 10^{-3} and are omitted for clarity. Results are averaged over 32 runs. **B)** Approximation of the reward probabilities learned by the continuous Bayesian Hebb rule after 2000 training trials. The learned approximation (dashed line) is very close to the true reward distribution (gray solid line).

3.7 Discussion

3.7.1 Summary and open problems

We have proposed in this article a simple neural network architecture for learning and decision making, which makes use of two learning processes that operate on two different time scales. We assume that generic dependencies among sensory input variables or features, or in other words, the factors of the underlying Bayesian network, are detected on a larger time scale, and that combinations of conditionally dependent input features are presented to the decision stage through sparse population coding. We have shown that on the basis of such preprocessing, the optimal policy can be represented as a WTA operation applied to weighted sums, and the corresponding weights can be learnt very fast. In fact, we have shown that a very simple Hebbian learning rule (the reward-modulated Bayesian Hebb rule) can integrate information from past experience in a close to optimal way. The models that we presented and analyzed are biologically plausible and arguably minimal with regard to their complexity, but nevertheless can be shown to asymptotically approximate theoretically optimal performance. All information from past experience is stored in synaptic weights of simple linear neuron models, and can therefore immediately be used for online decision making. In contrast to other learning rules that have previously been proposed for modeling animal learning — such as the Rescorla-Wagner rule [184, 234], the perceptron learning rule, or learning rules based on the Kalman-filter model [40, 216] — this new learning rule is a truly Hebbian learning rule. Its weight updates depend on the current pre- and postsynaptic activity, as well as on a third signal [14] that contains information about success or failure of the currently selected decision, but not on the current values of the other weights (or the resulting weighted sums of input variables). All information required for the weight update is therefore available locally at the synapse.

A major advantage of the local nature of purely Hebbian learning rules is that synapses can be removed or added to a neuron, without changing the target weights of the other synapses. One can therefore view the reward-modulated Bayesian Hebb rule as a candidate for learning in self-organizing organisms with developing neural structure. Assume, for example, that an input variable x_{new} is added, and the population code is appropriately modified. Then all weights belonging to factors in the factor graph that are not connected to x_{new} are unaffected, and can still be used for decision

making. Removal or addition of single weights does however affect the decision making process, if the resulting population code does not match either the SP or GP encoding.

The Bayesian Hebb rule is one of very few online learning rules that admit a rigorous theoretical analysis of their convergence properties. We have shown that the theoretically optimal values of the weights are fixed point attractors for expected weight changes (see Figure 3.2B). This implies in particular, that learning cannot get stuck in local minima of some loss function. In fact, one can easily show that the expected weight updates give rise to an exponentially fast contracting dynamical system in weight space. Hence, this learning process falls into the theoretical framework of *contracting systems*, proposed by [142]. According to this theory, this learning process can therefore be combined with other adaptive processes that also exhibit a contracting dynamics of adaptive parameters. Their theory guarantees that the resulting hybrid learning system will also converge.

We have also considered in section 3.3 a computationally simpler linear version of the Bayesian Hebb rule. Although this rule is only an approximation to the Bayesian Hebb rule, and theoretical convergence results are weaker (see the discussion in Section 3.3.1), we have shown that it performs almost equally well in a large number of complex decision making tasks (see Figures 3.6, 3.7, 3.11). The linear Bayesian Hebb rule is similar to well-known mathematical models for Hebbian learning, and may therefore provide a new interpretation of these learning rules as approximations to more complex plasticity mechanisms

In this article we have studied the scenario of online reward-based learning of decision making with multiple alternatives from stochastic rewards and input signals, which is important for fields like operant conditioning or reinforcement learning. In section 3.2.2 we have shown analytically and empirically (Figure 3.2A) that the Bayesian Hebb rule achieves near optimal learning in terms of learning speed, and asymptotically approaches the optimal policy for the given preprocessing mechanism. We have supported this theoretical prediction through a variety of computer simulations of decision tasks (see Figures 3.5, 3.8, 3.10). The resulting higher learning speed is particularly interesting in our context of reward-based learning, where most learning algorithms are too slow to be applicable to real-world problems. Hence the contribution of this article can be seen as another step in the program to speed up reinforcement learning by mak-

ing near optimal use of previous experience. We have shown in section 3.5.2 that this approach can also be applied to non-stationary distributions of inputs and rewards.

The question, how the brain forms decisions that involve more than two alternatives, is one of the most important open research problems [81]. For binary decisions, Wald's sequential probability ratio test [229] provides a theoretically optimal tool for learning and decision making from limited evidence. In this case it is sufficient to update a single decision variable, and compare it to a threshold value. For problems with more than 2 alternatives it is unclear whether an optimal test exists, and tests that guarantee asymptotic optimality, such as the method developed by [55] become much more complex (see [92] for a possible neural implementation). In this article we have studied a simpler network model, which does not select actions optimally in the sense of sequential analysis. It converges asymptotically to an optimal policy, and uses heuristic strategies for choosing actions during learning. We have analyzed a model that is based on the Winner-Take-All (WTA) operation, and directly uses the learned weights for the evaluation of actions. We have shown that if WTA is applied to several linear neurons, each of which learns via the Bayesian Hebb rule to approximate the log-odd of receiving a reward for an associated action (see Figure 3.1), our simple model can handle the case of more than two decision alternatives without any extra effort (see sections 3.2 and 3.3 for the theoretical analysis and Figures 3.5, 3.6, 3.8, 3.10 for empirical tasks).

WTA-circuits are of interest in the context of neural network models for action selection, since it has been suggested that generic cortical microcircuits implement a soft version of WTA-circuits (where $z_a > 0$ also for the runner-ups in the competition among the L_a), see [52]. This view is supported by the anatomical observation that the output cells (pyramidal neurons) of cortical microcircuits are subject to lateral inhibition (each pyramidal neuron excites inhibitory interneurons that target other pyramidal neurons). It is also supported by the physiological observation that simultaneous activation of very large numbers of sensory neurons (for example in the retina) is transformed through cortical processing into sparse activity of neurons in higher sensory areas (e.g., area IT). Consequently, WTA-circuits have become a primary target for the design of neurally inspired electronic hardware [95, 157].

The components of our neural network model (Figure 3.1) have substantial experimental and theoretical support. Hebbian learning, and the use of weighted sums for decision making [190] is clearly feasible for biological neurons. The other essential

ingredient of our model for reward-based learning of decision making is a suitable preprocessing of variables $\mathbf{x}$ (typically representing sensory inputs) that form the evidence on which a decision has to be based in a single trial. Our model requires a sparse population coding of the values of these variables (both for variables with discrete and for variables with continuous values, see section 3.6). Sparse encodings [167], or population codes are common models for coding strategies of the brain, and experimental evidence for the existence of such codes has been found in various brain areas of different species [see e.g. 71, 165, 179].

Furthermore, in the case of conditioned dependencies among variables our model assumes that there exists a population coding for "complex features" (reminiscent of neural codes reported for example for visual areas V2 and IT), i.e. for combinations of variables (see Figure 3.4 for an example). Hence, our simple neural network model for learning decision making entails concrete predictions for the computational strategies, neural codes, and learning mechanisms in those cortical areas that provide information about sensory inputs in a highly processed form to other cortical areas where decisions are made. It proposes that those subgroups of sensory variables (from the same or different sensory modalities) that have statistical dependencies, such as those represented by a factor graph [130], are brought together in some cortical microcircuits, and that projection neurons from these cortical microcircuits each assume a high firing rate for a particular combination of values of these variables (thereby mimicking the output variables y_i of our general preprocessing, see section 3.4.1).

This link of factor graph theory and experimentally observed population codes provides a novel view on the potential role of sensory areas that provide input to higher decision making stages in the brain. The proposed preprocessing has the advantage of relieving the subsequent decision stage from complex computations (such as belief propagation via message passing) and nonlinear learning devices. In fact, it enables the decision stage to use only linear operations in conjunction with WTA. It also enables the decision stage to accumulate evidence from history through the very simple and robust Hebbian learning processes that were discussed in this article.

In this article we assume that the graph structure of the factor graph is known, which is a very common assumption for parameter learning algorithms in graphical models [see e.g. 111, 156]. The evolution of preprocessing circuits is obviously a complex process, and the design of learning algorithms that generate such preprocessing of

sensory inputs is an interesting open problem. Testing variables for (conditional) dependence is perhaps a less formidable problem for a neural network than it may appear on first sight, provided one assumes that numerous autonomous learning processes try to predict each variable in terms of others. Dependencies among the variables exist, and can in principle be found autonomously by this process, whenever such prediction learning turns out to be successful. As mentioned above, such relationships between input signals may be learned on much longer time scales than decision strategies, which require very fast adaptation.

Other obvious open problems that arise from our model are whether it can be implemented with spiking neurons, and whether there exist relationships between the theoretically optimal reward-modulated Bayesian Hebb rule and concrete heterosynaptic learning mechanisms of biological synapses such as those discussed in [14]. Another open problem concerns a possible extension of our model to rewards signals with more than two values, to third signals that represent predictions of rewards, and to reward based learning in continuous time.

Altogether our simple neural network model for learning decision making has shown that this problem is in some aspects less difficult than it may appear on first sight. It remains to be explored whether biological neural systems have adopted related implementation strategies, or have found even simpler solutions to this problem.

3.7.2 Related Work

3.7.2.1 Models for Decision Making

The study of decision making in biological systems dates back to the classical experiments by Pavlov, in which dogs learned associations between cues and rewards. On the other hand, operant or instrumental conditioning is concerned with associations between actions and rewards, and how behavior is modified through reward and punishment. The goal is to learn a policy, i.e. a way to select actions near-optimally in response to environmental stimuli. According to [214], biological organisms first transform sensory input into decision related variables, e.g. value representations in area LIP for visual discrimination tasks in monkeys [232]. An unknown computational mechanism maps the values of these variables to the probability of reward for executing various actions, which then leads to a motor response. An actor-critic model is assumed,

in which the actor and the critic are two modules that operate with a common reward currency. The critic adapts the value of every action to the perceived reward probabilities, thereby altering the decision transformation, which the actor uses to choose actions. An example for models of instrumental conditioning is the experiment of [152], in which the behavior of a foraging bee is simulated with a neural network model and a suitable learning rule (a variation of the Rescorla-Wagner rule). [230] has described a recurrent cortical network model, which uses feedback and winner-take-all mechanisms to integrate information in visual discrimination tasks with two possible outcomes. [92] have presented a model for optimal decision making with multiple actions, which models the functionality of the basal ganglia. Further neural network models for decision making have been reviewed in [195].

3.7.2.2 Learning Rules for Decision Making

The classical model for learning associations of stimuli, actions, and rewards is the Rescorla-Wagner rule [184]. It was the first mathematical model for learning that could explain most of the effects observed in animal behavior studies. In particular it was able to explain reactions based on combinations of stimuli. Reward associations for many conditioning paradigms, such as e.g. partial reinforcement, inhibitory conditioning, or extinction can be learned by the Rescorla-Wagner rule (and also by the Bayesian Hebb rule). The associative model of the Rescorla-Wagner rule represents the predicted amount of reward as a weighted sum of stimuli, and weights are updated using the difference between the predicted and the actually received reward (see (3.26)). The Rescorla-Wagner rule is therefore not a strictly Hebbian learning rule, because this error signal, rather than the activation of the post-synaptic neuron is required for the update. Studies by Schultz et al. have however indicated that such an error signal may be available in the form of the neuromodulator dopamine [204].

Learning rules that minimize prediction errors were also useful to explain blocking phenomena in conditioning [38]. However, some observed effects like backward blocking — an established reward association is unlearned, because another stimulus sufficiently explains the occurrence of rewards — can neither be sufficiently captured by the Rescorla-Wagner rule, nor by the Bayesian Hebb rule. The reason for this is that weights in these models can only be reduced, if unrewarded trials are observed (which is not the case in the backward blocking paradigm). Algorithms that specifically address

learning of reward associations in the backward blocking scenario are based on Kalman filter models for conditioning [216]. [41] argue that in addition to error correction, it is necessary to model the uncertainty in the parameter estimates during learning, and neuromodulators like acetylcholine or norepinephrine could signal such uncertainty in biological systems [233]. An artificial recurrent neural network model, which approximates the Kalman filter estimates of reward associations for backward blocking was presented by [40]. A different learning mechanism is suggested by [85], who argue that phenomena like backward blocking could also be modeled by learning changes in the causal structure of the problem, rather than by learning new reward associations.

The mathematical problem of learning optimal action selection is also well-studied in the field of reinforcement learning (RL) [217]. Typical RL algorithms learn value- or Q-functions, which estimate the expected reward resulting from the execution of action a in state $\mathbf{x}$. The goal of RL is to converge to optimal policies, which select for every state those actions that maximize the expected reward (typically a discounted long-term reward for sequential decision problems). Classical RL algorithms do not directly aim at maximizing the online performance, i.e. the amount of reward obtained during learning, but typically employ some heuristics to tackle the exploration-exploitation dilemma. This dilemma concerns the trade-off of online performance (exploitation) and exploration of unseen parts of the state- and action space in order to improve the final policy. More recently the problem of optimizing online performance has attracted more attention in the RL literature [e.g. 9, 11, 115]. Asymptotic convergence of RL algorithms to the optimal policy can only be guaranteed for discrete environments, if action values are stored in look-up tables with one entry for every combination of state and action. Such tabular representations are biologically not realistic, and for computers the memory requirements are too large for most real-world applications. Value functions are therefore approximated, but convergence results exist only for a limited number of approximation schemes [17].

Using Bayesian inference for action selection in uncertain environments was e.g. studied by [8], and [227]. They consider the problem of planning action sequences of fixed length for partially observable Markov decision processes with one or more fixed goal states. The dynamics of the environment are initially unknown. The learning part uses frequency counters to update conditional probabilities for transition and reward models. Planning is reduced to Bayesian inference in graphical models based on the

learned parameters, which is computed with standard algorithms, like e.g. belief propagation or the junction tree algorithm [22]. The posterior over actions, given that start and goal state are fixed, is computed and the maximally likely sequence of actions (and intermediate states in [227]) is selected. This approach is conceptually quite different from our approach, since our approach does not learn sequences of actions, and does not require a defined goal-state. The learned parameters in our model (the weights $w_{a,i}$) are not auxiliary variables, but are directly used in the decision making process. Furthermore, our approach only requires very basic and apparently biologically feasible mechanisms like Hebbian learning, weighted summations, and winner-take-all. Implementing full Bayesian inference is a much more difficult process, for which it is not clear how the brain can achieve it efficiently, although some models have been proposed (e.g. [45, 182]). [126, 133, 134] and [197] have studied various learning rules (although not in a reinforcement learning context) that approximate optimal Bayesian inference. The learning rules differ from the Bayesian Hebb rule that was introduced in this article primarily by fact that they require auxiliary counters for storing evidence from past experience.

3.7.2.3 Analogies to recent experimental studies of decision making in primates

Recent experimental results by [232] have shown that the previous experience of macaque monkeys in probabilistic decision tasks is represented by the firing rates of neurons in area LIP in the form of the log-likelihood ratio of receiving a reward for a particular action a in response to a stimulus $\mathbf{x}$, like in equation (3.1) of our framework. In their experiment a monkey had to choose at each trial between two possible actions. It could choose to move the eyes either towards a red target R ($a = R$) or a green target G ($a = G$). The probability that a reward was received at either choice depended on four visual input stimuli $\mathbf{x} = (x_1, x_2, x_3, x_4)$ that had been shown at the beginning of the trial. Every stimulus $x_k, k = 1, \ldots, 4$, was one shape s_j out of a set of ten possibilities $\{s_1, \ldots, s_{10}\}$ and had an associated weight $\omega_k = \omega(s_j)$, which had been defined by the experimenter. The log-odd of obtaining a reward was equal to the sum of $\omega_1, \ldots, \omega_4$:

$$\log \frac{p(r = 1|\mathbf{x}, a = R)}{p(r = 1|\mathbf{x}, a = G)} = \sum_{k=1}^{4} \omega_k \quad . \tag{3.34}$$

The monkey thus had to combine the evidence from four visual stimuli to optimize its action selection behavior. It also had to find out that reward probabilities only depended on the presented shapes, but not on the order or location in which they were presented. A reward was assigned before the trial to one of the targets according to the distribution (3.34).

One can easily model this task in our framework, using a simple population code $\mathbf{y} = \mathbf{\Phi}(\mathbf{x})$ as in (3.18), where the stimulus $\mathbf{x}$ was encoded by a 40-dimensional binary vector $\mathbf{y}$ with exactly $m = 4$ inputs being 1. The positions of the 1's corresponded to the four visual shapes that were shown during a trial. The log-odd of obtaining reward with action $a = R$ can then be written as a weighted sum

$$\log \frac{p(r = 1|\mathbf{y}, a = R)}{p(r = 0|\mathbf{y}, a = R)} = \sum_{i=1}^{40} w_i^* y_i \quad , \tag{3.35}$$

with

$$w_i^* \quad = \quad \log \frac{p(r = 1|y_i = 1, a = R)}{p(r = 0|y_i = 1, a = R)} \quad . \tag{3.36}$$

Due to the symmetry of the task (reward is either at R or G), the log-odds in (3.34) and (3.35) are equivalent. The weights w_i can be learned with an efficient version of the reward-modulated Bayesian Hebb rule (3.8), which takes this symmetry into account. The equilibrium w_i^* of weight w_i under this slightly modified rule is then exactly at the desired value (3.36). We simulated this task, using a learner with the reward-modulated Bayesian Hebb rule and a $1/N_i$ learning rate for every weight. Figure 3.11A shows that this task can be successfully learned both by the exact reward-modulated Bayesian Hebb rule (3.8) and the linear approximation (3.12). The learning rules learn as fast as the non-Hebbian Rescorla-Wagner rule (3.26), and their performance is close to the theoretical optimum after 1000 training trials. Furthermore Figures 3.11B and C show that the intermediate and final policies resemble the behavior that was reported for two monkeys in [232].

The experimental data of [232] are consistent with the assumption that monkeys apply a WTA-operation to the log-likelihood ratios

$$L_a = \log \frac{p(r = 1|\mathbf{x}, a)}{p(r = 0|\mathbf{x}, a)} \quad ,$$

which are, according to their model, represented through firing rates of neurons in area LIP. It is not known, which values are represented by the firing rates y_i of the

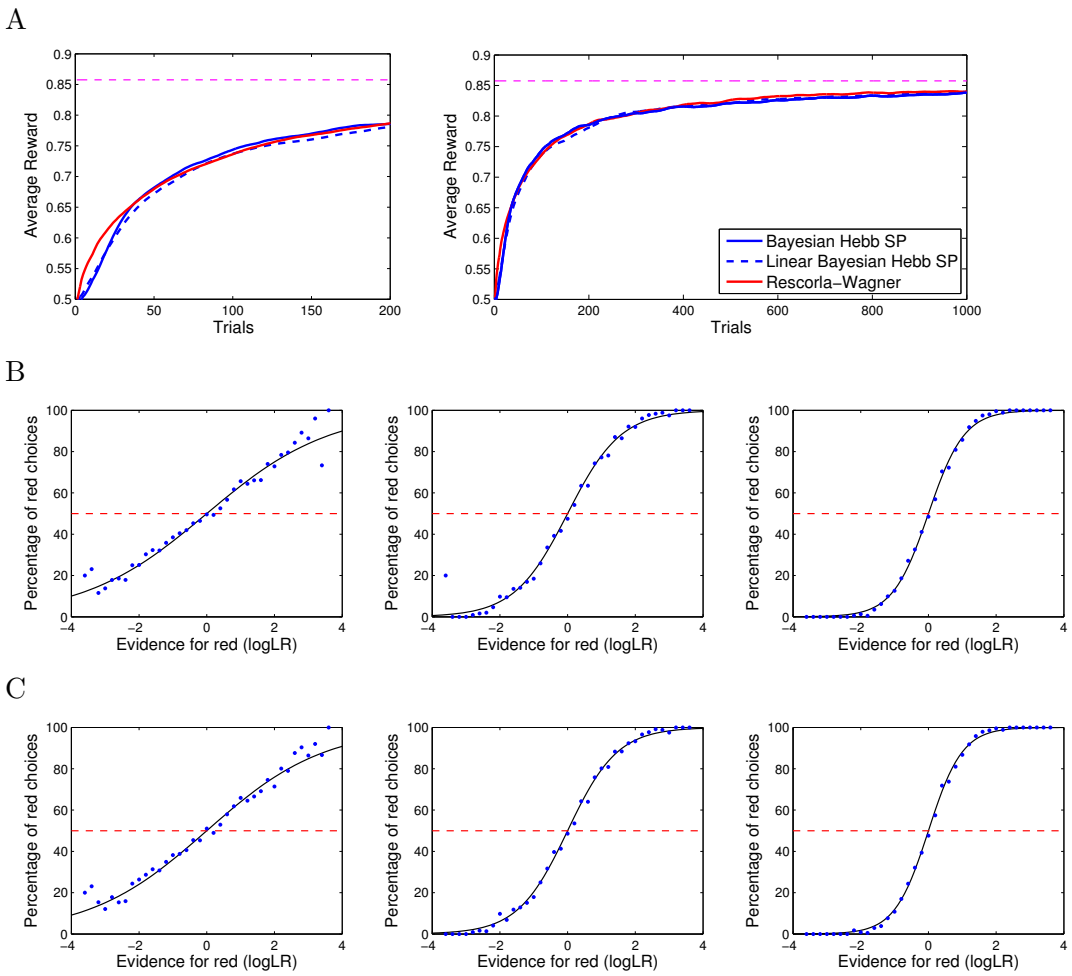

Figure 3.11: Performance of the reward-modulated Bayesian Hebb rule in the model for the conditioning task by Yang and Shadlen (see [232] for details). **A)** The reward-modulated Bayesian Hebb rule learns as fast as the non-Hebbian Rescorla-Wagner rule (curves result from averaging over 32 repetitions of the experiment, where the average reward was measured on 500 independent test trials). The horizontal dashed line reflects the theoretically best possible performance. Error bars are in the range of 10^{-2} and are omitted for clarity. **B, C)** Action selection policies (greedy policy according to (3.3)) resulting from the model using the exact Bayesian Hebb rule (3.8) (**B**) or the linear Bayesian Hebb rule (3.12) (**C**) after 100 (left), 500 (middle), and 1000 (right) trials, fitted by sigmoidal curves (results are from 32 repetitions of the experiment, where the behavior was measured on 1000 independent test trials). The policies represented by the left and right panels are qualitatively similar to the policies adopted by monkeys H and J in the experiments by [232] after learning (see Figure 1b in [232]).

presynaptic neurons of these neurons. In our simple model we model the neurons within the WTA circuit as linear neurons, and assume that their output L_a can be written as a linear sum $L_a = \sum_{i=0}^{n} w_{a,i}\ y_i$ of variables y_i that represent a population coding of the sensory input $\mathbf{x}$. As we have shown in section 3.4, if this population coding is chosen in a suitable way, the true reward log-odd $\log \frac{p(r=1|\mathbf{x},a)}{p(r=0|\mathbf{x},a)}$ can in fact be written as such weighted sum. Hence our theoretical framework makes concrete predictions about the nature of the transformation of raw sensory inputs $\mathbf{x}$ to inputs $\mathbf{y}$ for higher brain areas that select suitable responses. The required weights $w_{a,i}$ can be learnt by the reward-modulated Bayesian Hebb rule, and a linear Poisson neuron whose weights are updated according to this rule will adapt for each trial a firing rate proportional to the log-likelihood ratio $\log \frac{p(a=R|\mathbf{x})}{p(a=G|\mathbf{x})}$. This response matches that of the neurons in area LIP shown in Figure 2c and 3b of [232].[1]

The Bayesian Hebb rule provides an arguably minimal model for the biological data of [232]. One difference between their results and our model is that learning is much faster in our model. This could be explained by the fact that many aspects of the probabilistic decision task of [232] - e.g. the fact that the reward policy was stationary, the fact that the reward probabilities did not depend on the order of appearance, or the spatial location of the shown icons, and the fact that reward probabilities did not depend on any other aspects that the monkeys had perceived before or during a session - also had to be learned by the monkeys, whereas they were assumed as given in our model. Learning of these invariances and symmetries was actually quite hard in the set-up of [232] since rewards were given stochastically, rather than by deterministic laws (note that even many humans believe to "learn" various misleading reward-predictors while gambling for a long time in the lottery or casinos). An interesting open question is whether reward-based learning of decision making by humans or animals can approach the learning speed of the Bayesian Hebb rule when such differences between the learning tasks of the living organisms and the mathematical model have been removed.

3.7.3 Conclusion

We have demonstrated the functionality of a simple neural network model for learning of asymptotically optimal action selection, which uses only biologically plausible

[1] Note that the optimal weights w_i^* are equal to the weights $\omega_k = \omega(s_j)$ that were assigned to the different visual shapes s_j.

mechanisms such as reward-modulated Hebbian learning, sparse population coding, and winner-take-all computations. Furthermore we have shown that on the basis of a suitable preprocessing that takes dependencies among salient variables into account, a very simple Hebbian learning rule can converge towards optimal policies extremely fast. On the side, our approach offers concrete processing goals for brain areas that integrate multi-modal sensory input, in order to facilitate learning and decision making in higher brain areas. Empirical results have confirmed that the new reward-modulated Bayesian Hebb rule, and an even simpler linear approximation to it, compare favorably to well-known non-Hebbian learning rules for action-selection tasks. Our results suggest that learning and decision making under uncertainty can be implemented very efficiently in biological neural systems.

3.8 Proofs

3.8.1 Convergence proofs for the Bayesian Hebb rule

We assume that $p(r|\mathbf{y}, a)$, the reward probability conditioned on the current input and action, is stationary, and $p(y_i = 1, a) > 0$ for all $a \in A$ and $i \in \{1, \ldots, n\}$. Apart from the latter assumption, the equilibrium is independent of the exploration policy $\pi(\mathbf{x}, a)$. The constraint on $p(y_i = 1, a)$ means that all values of all input variables must have a non-zero probability in the input-distribution, and every action must have a non-zero probability of being tried out. If $p(y_i = 1, a) = 0$ for some y_i and a, then such trials are never encountered, and no meaningful weight $w_{a,i}$ can be learned.

Since updates of $w_{a,i}$ in (3.8) are only made when a is executed and $y_i = 1$, one can write

$$\begin{aligned}
E[\Delta w_{a,i}] = 0 \quad &\Leftrightarrow \quad p(r = 1|y_i = 1, a) \cdot \eta \cdot (1 + e^{-w_{a,i}}) - \\
&\quad -p(r = 0|y_i = 1, a) \cdot \eta \cdot (1 + e^{w_{a,i}}) = 0 \\
&\Leftrightarrow \quad \frac{1 + e^{w_{a,i}}}{1 + e^{-w_{a,i}}} = \frac{p(r = 1|y_i = 1, a)}{p(r = 0|y_i = 1, a)} \\
&\Leftrightarrow \quad e^{w_{a,i}} = \frac{p(r = 1|y_i = 1, a)}{p(r = 0|y_i = 1, a)} \\
&\Leftrightarrow \quad w_{a,i} = \log \frac{p(r = 1|y_i = 1, a)}{p(r = 0|y_i = 1, a)}.
\end{aligned}$$

The above is a chain of equivalence transformations, therefore $w^*_{a,i} = log\frac{p(r=1|y_i=1,a)}{p(r=0|y_i=1,a)}$ is the only equilibrium value of rule (3.8).

One can also show that the expected update of weights $w_{a,i}$ is always in the right direction:

$$\begin{aligned}
&E[\Delta w_{a,i}]|_{w^*_{a,i}+2\epsilon} = E[\Delta w_{a,i}]|_{w^*_{a,i}+2\epsilon} - E[\Delta w_{a,i}]|_{w^*_{a,i}} \\
&\propto p(r{=}1|y_i = 1, a)e^{-w^*_{a,i}}(e^{-2\epsilon} - 1) - p(r{=}0|y_i = 1, a)e^{w^*_{a,i}}(e^{2\epsilon} - 1) \\
&= p(r{=}0|y_i = 1, a)(e^{-2\epsilon} - 1) - p(r{=}1|y_i = 1, a)(e^{2\epsilon} - 1) \\
&= \left[(p(r{=}0|y_i = 1, a)e^{-\epsilon} + p(r{=}1|y_i = 1, a)e^{\epsilon}\right] (e^{-\epsilon} - e^{\epsilon}). \qquad (3.37)
\end{aligned}$$

The first term in (3.37) is always positive, and from the last term in (3.37) one can see that whenever $w_{a,i} > w^*_{a,i}$, i.e. $\epsilon > 0$, the expected change of $w_{a,i}$ is negative, and positive if $\epsilon < 0$. The expected change of weights is therefore always in the direction of the optimal weight, and the initial weight values or pertubations of the weights decay exponentially fast. Furthermore, trajectories of weights that start at different initial values converge exponentially fast. Hence the resulting weight dynamics is contracting in the sense of [142].

3.8.2 Convergence proof for the Linear Bayesian Hebb rule

The expected update of the linear Bayesian Hebb rule (3.12) vanishes when

$$\begin{aligned}
E[\Delta w_{a,i}] = 0 \quad &\Leftrightarrow \quad p(r = 1|y_i = 1, a) \cdot \eta \cdot (2 - w_{a,i}) - p(r = 0|y_i = 1, a) \cdot \eta \cdot (2 + w_{a,i}) = 0 \\
&\Leftrightarrow \quad 2(p(r = 1|y_i = 1, a) - p(r = 0|y_i = 1, a)) = \\
&\qquad = w_{a,i} \cdot (p(r = 1|y_i = 1, a) + p(r = 0|y_i = 1, a)) = w_{a,i} \\
&\Leftrightarrow \quad w_{a,i} = 2(p(r = 1|y_i = 1, a) - 1 + p(r = 1|y_i = 1, a)) \\
&\Leftrightarrow \quad w_{a,i} = -2 + 4 \cdot p(r = 1|y_i = 1, a) \quad .
\end{aligned}$$

We have used here that the reward is binary, and so

$$p(r = 0|y_i = 1, a) + p(r = 1|y_i = 1, a) = 1 \quad .$$

The above is a chain of equivalence transformations, so $w^+_{a,i} = -2+4\cdot p(r = 1|y_i = 1, a)$ is the only equilibrium value of (3.12).

3.8.3 Derivation of the population code for the Naive Bayes case

From the Naive Bayes assumption we know that

$$\frac{p(r=1|\mathbf{x},a)}{p(r=0|\mathbf{x},a)} = \frac{p(r=1|a)}{p(r=0|a)} \prod_{k=1}^{m} \frac{p(x_k|r=1,a)}{p(x_k|r=0,a)} \,. \tag{3.38}$$

Each discrete conditional distribution $p(x_k|r,a)$ for a fixed action a and a fixed value of r is fully described by m_k probability values, one for each possible value of x_k, and can be written in the form

$$p(x_k|r,a) = p(x_k=1|r,a)^{I(x_k=1)} \cdot p(x_k=2|r,a)^{I(x_k=2)} \cdot \ldots \cdot p(x_k=m_k|r,a)^{I(x_k=m_k)},$$

where the indicator function I is defined as $I(\texttt{true}) = 1$ and $I(\texttt{false}) = 0$. With this notation (3.38) can be rewritten as

$$\begin{aligned}
\frac{p(r=1|\mathbf{x},a)}{p(r=0|\mathbf{x},a)} &= \frac{p(r=1|a)}{p(r=0|a)} \prod_{k=1}^{m} \prod_{j=1}^{m_k} \left(\frac{p(x_k=j|r=1,a)}{p(x_k=j|r=0,a)} \right)^{I(x_k=j)} \\
&= \frac{p(r=1|a)}{p(r=0|a)} \prod_{k=1}^{m} \prod_{j=1}^{m_k} \left(\frac{p(r=1|x_k=j,a)}{p(r=0|x_k=j,a)} \cdot \frac{p(r=0|a)}{p(r=1|a)} \right)^{I(x_k=j)} \\
&= \left(\frac{p(r=1|a)}{p(r=0|a)} \right)^{1-m} \prod_{k=1}^{m} \prod_{j=1}^{m_k} \left(\frac{p(r=1|x_k=j,a)}{p(r=0|x_k=j,a)} \right)^{I(x_k=j)} \,.
\end{aligned} \tag{3.39}$$

3.8.4 Convergence proof for the Continuous Bayesian Hebb rule

The equilibrium of the continuous Bayesian Hebb rule (3.31) is reached when the expected update $E[\Delta w_{a,i}]$ vanishes:

$$\begin{aligned}
E[\Delta w_{a,i}] = 0 \quad \Leftrightarrow \quad & (1+e^{-w_{a,i}}) \int_X y_i(\mathbf{x}) \; p(r=1,\mathbf{x}|a) \; d\mathbf{x} \\
& -(1+e^{w_{a,i}}) \int_X y_i(\mathbf{x}) \; p(r=0,\mathbf{x}|a) \; d\mathbf{x} = 0
\end{aligned}$$

We now use the interpretation of $y_i(\mathbf{x})$ as $p(\tilde{y}_i = 1|\mathbf{x})$. Since the virtual population code feature $\tilde{y}_i$ depends only on $\mathbf{x}$, but not on r, one can assume that r and $\tilde{y}_i$ are conditionally independent given $\mathbf{x}$, i.e.

$$p(r,\tilde{y}_i|\mathbf{x},a) = p(r|\mathbf{x},a) \cdot p(\tilde{y}_i|\mathbf{x}) \quad .$$

This assumption, and simple transformations using basic laws of probability lead to

$$
\begin{aligned}
E[\Delta w_{a,i}] = 0 \quad &\Leftrightarrow \quad \frac{1+e^{w_{a,i}}}{1+e^{-w_{a,i}}} = \frac{\int_X p(\tilde{y}_i = 1|\mathbf{x})\ p(r=1|\mathbf{x},a)\ p(\mathbf{x}|a)\ d\mathbf{x}}{\int_X p(\tilde{y}_i = 1|\mathbf{x})\ p(r=0|\mathbf{x},a)\ p(\mathbf{x}|a)\ d\mathbf{x}} \\
&\Leftrightarrow \quad e^{w_{a,i}} = \frac{\int_X p(\tilde{y}_i = 1, r=1|\mathbf{x},a)\ p(\mathbf{x}|a)\ d\mathbf{x}}{\int_X p(\tilde{y}_i = 1, r=0|\mathbf{x},a)\ p(\mathbf{x}|a)\ d\mathbf{x}} \\
&\Leftrightarrow \quad e^{w_{a,i}} = \frac{\int_X p(\tilde{y}_i = 1, r=1, \mathbf{x}|a) d\mathbf{x}}{\int_X p(\tilde{y}_i = 1, r=0, \mathbf{x}|a) d\mathbf{x}} \\
&\Leftrightarrow \quad e^{w_{a,i}} = \frac{p(\tilde{y}_i = 1, r=1|a)}{p(\tilde{y}_i = 1, r=0|a)} \\
&\Leftrightarrow \quad e^{w_{a,i}} = \frac{p(r=1|\tilde{y}_i = 1, a)}{p(r=0|\tilde{y}_i = 1, a)} \\
&\Leftrightarrow \quad w_{a,i} = \log \frac{p(r=1|\tilde{y}_i = 1, a)}{p(r=0|\tilde{y}_i = 1, a)} \quad .
\end{aligned}
$$

3.8.5 Performance of the Rescorla-Wagner rule with preprocessing

The performance of the Rescorla-Wagner rule (3.26) can be improved by preprocessing input signals before the learning rule is applied. Figure 3.12 shows the average reward for the two tasks studied in Figure 3.5 (with 2 binary inputs and 4 actions), and Figure 3.8 (with 100 binary inputs and 10 actions). When the Rescorla-Wagner rule (3.26) was applied to simple population coding (*SP*) or to generalized preprocessing (*GP*), it learned faster and converged to better policies, although the performance of the Bayesian Hebb rule was mostly superior. These results suggest that the preprocessing methods presented in section 3.4, could also be beneficial for other learning mechanisms. The Augmented Rescorla-Wagner rule [234] uses a preprocessing mechanism similar to GP, but it did not perform better for the experiments in this article.

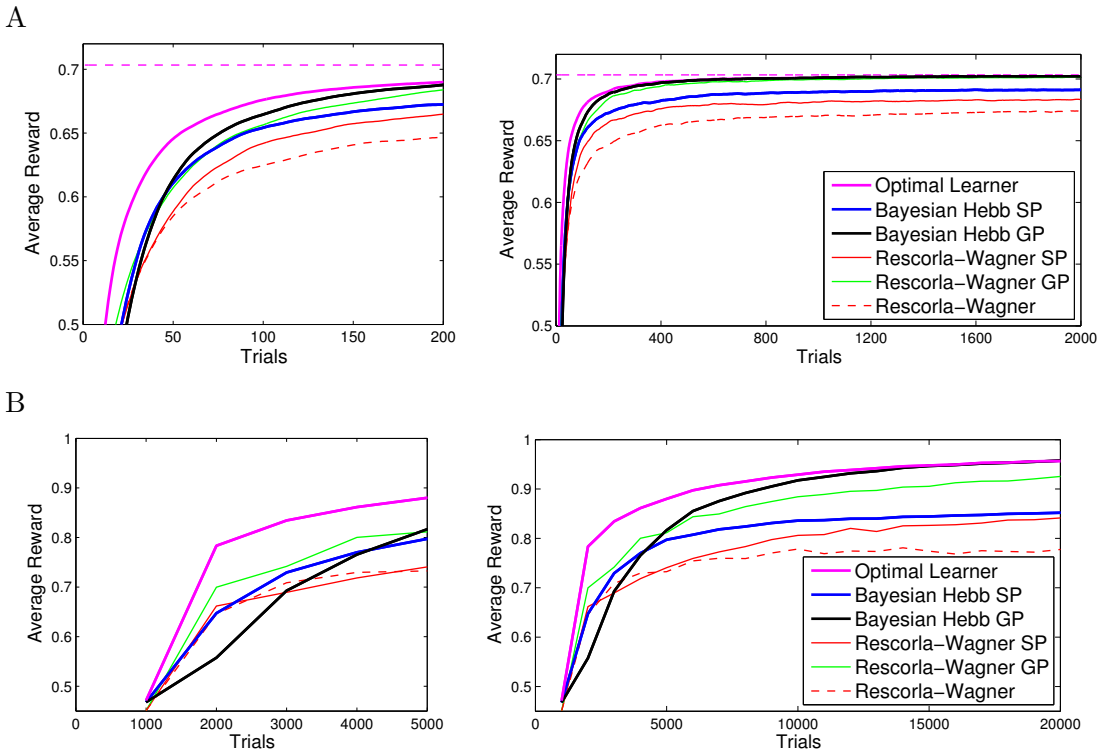

Figure 3.12: The performance of the Rescorla-Wagner rule can be improved by preprocessing input signals. The Rescorla-Wagner rule was applied to preprocessed inputs using simple population coding (*Rescorla-Wagner SP*), or generalized preprocessing (*Rescorla-Wagner GP*). The Rescorla-Wagner rule with preprocessing generally learned faster, and converged to better policies than the classical Rescorla-Wagner rule. **A)** Performance for the same 4-action tasks with 2 binary input variables as in Figure 3.5. **B)** Performance in the same 10-action tasks with 100 binary input variables as in Figure 3.8.

4

Bayesian Computation Emerges in Generic Cortical Microcircuits through Spike-Timing-Dependent Plasticity

Abstract

The principles by which networks of neurons compute, and how spike-timing dependent plasticity (STDP) of synaptic weights generates and maintains their computational function, are unknown. Preceding work has shown that soft winner-take-all (WTA) circuits, where pyramidal neurons inhibit each other via interneurons, are a common motif of cortical microcircuits. We show through theoretical analysis and computer simulations that Bayesian computation is induced in these network motifs through STDP in combination with activity-dependent changes in the excitability of neurons. The fundamental components of this emergent Bayesian computation are priors that result from adaptation of neuronal excitability and implicit generative models for hidden causes that are created in the synaptic weights through STDP. In fact, a surprising result is that STDP is able to approximate a powerful principle for fitting such implicit generative models to high-dimensional spike inputs: Expectation Maximization. Our results suggest that the experimentally observed spontaneous activity and trial-to-trial variability of cortical neurons are essential features of their information processing

capability, since their functional role is to represent probability distributions rather than static neural codes. Furthermore it suggests networks of Bayesian computation modules as a new model for distributed information processing in the cortex.

Author Summary

How do neurons learn to extract information from their inputs, and perform meaningful computations? Neurons receive inputs as continuous streams of action potentials or "spikes" that arrive at thousands of synapses. The strength of these synapses - the synaptic weight - undergoes constant modification. It has been demonstrated in numerous experiments that this modification depends on the temporal order of spikes in the pre- and postsynaptic neuron, a rule known as STDP, but it has remained unclear, how this contributes to higher level functions in neural network architectures. In this paper we show that STDP induces in a commonly found connectivity motif in the cortex - a winner-take-all (WTA) network - autonomous, self-organized learning of probabilistic models of the input. The resulting function of the neural circuit is Bayesian computation on the input spike trains. Such unsupervised learning has previously been studied extensively on an abstract, algorithmical level. We show that STDP approximates one of the most powerful learning methods in machine learning, Expectation-Maximization (EM). In a series of computer simulations we demonstrate that this enables STDP in WTA circuits to solve complex learning tasks, reaching a performance level that surpasses previous uses of spiking neural networks.

4.1 Introduction

Numerous experimental data show that the brain applies principles of Bayesian inference for analyzing sensory stimuli, for reasoning and for producing adequate motor outputs [54, 86, 87, 127, 183]. Bayesian inference has been suggested as a mechanism for the important task of probabilistic perception [62], in which hidden causes (e.g. the categories of objects) that explain noisy and potentially ambiguous sensory inputs have to be inferred. This process requires the combination of prior beliefs about the availability of causes in the environment, and probabilistic generative models of likely sensory observations that result from any given cause. By Bayes Theorem, the result

of the inference process yields a *posterior* probability distribution over hidden causes that is computed by multiplying the *prior* probability with the *likelihood* of the sensory evidence for all possible causes. In this article we refer to the computation of posterior probabilities through a combination of probabilistic prior and likelihood models as Bayesian computation. It has previously been shown that priors and models that encode likelihoods of external stimuli for a given cause can be represented in the parameters of neural network models [62, 143]. However, in spite of the existing evidence that Bayesian computation is a primary information processing step in the brain, it has remained open how networks of neurons can acquire these priors and likelihood models, and how they combine them to arrive at posterior distributions of hidden causes.

The fundamental computational units of the brain, neurons and synapses, are well characterized. The synaptic connections are subject to various forms of plasticity, and recent experimental results have emphasized the role of STDP, which constantly modifies synaptic strengths (weights) in dependence of the difference between the firing times of the pre- and postsynaptic neurons (see [36, 60] for reviews). Functional consequences of STDP can resemble those of rate-based Hebbian models [209], but may also lead to the emergence of temporal coding [117] and rate-normalization [1, 118]. In addition, the excitability of neurons is modified through their firing activity [37]. Some hints about the organization of local computations in stereotypical columns or so-called cortical microcircuits [88] arises from data about the anatomical structure of these hypothesized basis computational modules of the brain. In particular, it has been observed that local ensembles of pyramidal neurons on layers 2/3 and layers 5/6 typically inhibit each other, via indirect synaptic connections involving inhibitory neurons [52]. These ubiquitous network motifs were called soft winner-take-all (WTA) circuits, and have been suggested as neural network models for implementing functions like nonlinear selection [52, 95], normalization [30], selective attention [108], decision making [158, 173], or as primitives for general purpose computation [144, 192].

A comprehensive theory that explains the emergence of computational function in WTA networks of spiking neurons through STDP has so far been lacking. We show in this article that STDP and adaptations of neural excitability are likely to provide the fundamental components of Bayesian computation in soft WTA circuits, yielding representations of posterior distributions for hidden causes of high-dimensional spike inputs through the firing probabilities of pyramidal neurons. This is shown in detail

for a simple, but very relevant feed-forward model of Bayesian inference, in which the distribution for a single hidden cause is inferred from the afferent spike trains. Our new theory thus describes how modules of soft WTA circuits can acquire and perform Bayesian computations to solve one of the fundamental tasks in perception, namely approximately inferring the category of an object from feed-forward input. Neural network models that can handle Bayesian inference in general graphical models, including bi-directional inference over arbitrary sets of random variables, explaining away effects, different statistical dependency models, or inference over time require more complex network architectures [27, 172], and are the topic of ongoing research. Such networks can be composed out of interconnected soft WTA circuits, which has been shown to be a powerful principle for designing neural networks that can solve arbitrary deterministic or stochastic computations [144, 172, 192]. Our theory can thus be seen as a first step towards learning the desired functionality of individual modules.

At the heart of this link between Bayesian computation and network motifs of cortical microcircuits lies a new theoretical insight on the micro-scale: If the STDP-induced changes in synaptic strength depend in a particular way on the current synaptic strength, STDP approximates for each synapse exponentially fast the conditional probability that the presynaptic neuron has fired just before the postsynaptic neuron (given that the postsynaptic neuron fires). This principle suggests that synaptic weights can be understood as conditional probabilities, and the ensemble of all weights of a neuron as a generative model for high-dimensional inputs that - after learning - causes it to fire with a probability that depends on how well its current input agrees with this generative model. The concept of a generative model is well known in theoretical neuroscience [99, 100], but it has so far primarily been applied in the context of an abstract non-spiking neural circuit architecture. In the Bayesian computations that we consider in this article, internal generative models are represented implicitly through the learned values of bottom-up weights in spiking soft-WTA circuits, and inference is carried out by neurons that integrate such synaptic inputs and compete for firing in a WTA circuit. In contrast to previous rate-based models for probabilistic inference [116, 198, 199] every spike in our model has a clear semantic interpretation: one spike indicates the instantaneous assignment of a certain value to an abstract variable represented by the firing neuron. In a Bayesian inference context, every input spike provides

evidence for an observed variable, whereas every output spike represents one stochastic sample from the posterior distribution over hidden causes encoded in the circuit.

We show that STDP is able to approximate the arguably most powerful known learning principle for creating these implicit generative models in the synaptic weights: Expectation Maximization (EM). The fact that STDP approximates EM is remarkable, since it is known from machine learning that EM can solve a fundamental chicken-and-egg problem of unsupervised learning systems [44]: To detect - without a teacher - hidden causes for complex input data, and to induce separate learning agents to specialize each on one of the hidden causes. The problem is that as long as the hidden causes are unknown to the learning system, it cannot tell the hidden units what to specialize on. EM is an iterative process, where initial guesses of hidden causes are applied to the current input (E-step) and successively improved (M-step), until a local maximum in the log-likelihood of the input data is reached. In fact, the basic idea of EM is so widely applicable and powerful that most state-of-the art machine learning approaches for discovering salient patterns or structures in real-world data without a human supervisor rely on some form of EM [22]. We show that in our spiking soft-WTA circuit each output spike can be viewed as an application of the E-step of EM. The subsequent modification of the synaptic weights between the presynaptic input neurons and the very neuron that has fired the postsynaptic spike according to STDP can be viewed as a move in the direction of the M-step of a stochastic online EM procedure. This procedure strives to create optimal internal models for high-dimensional spike inputs by maximizing their *log*-likelihood. We refer to this interpretation of the functional role of STDP in the context of spiking WTA circuits as **s**pike-based **E**xpectation **M**aximization (SEM).

This analysis gives rise to a new perspective of the computational role of local WTA circuits as parts of cortical microcircuits, and the role of STDP in such circuits: The fundamental computational operations of Bayesian computation (Bayes Theorem) for the inference of hidden causes from bottom-up input emerge in these local circuits through plasticity. The pyramidal neurons in the WTA circuit encode in their spikes samples from a posterior distribution over hidden causes for high-dimensional spike inputs. Inhibition in the WTA accounts for normalization [30], and in addition controls the rate at which samples are generated. The necessary multiplication of likelihoods (given by implicit generative models that are learned and encoded in their synaptic

weights) with simultaneously learned priors for hidden causes (in our model encoded in the neuronal excitability), does not require any extra computational machinery. Instead, it is automatically carried out (on the log scale) through linear features of standard neuron models. We demonstrate the emergent computational capability of these self-organizing modules for Bayesian computation through computer simulations. In fact, it turns out that a resulting configuration of networks of spiking neurons can solve demanding computational tasks, such as the discovery of prototypes for handwritten digits without any supervision. We also show that these emergent Bayesian computation modules are able to discover, and communicate through a sparse output spike code, repeating spatio-temporal patterns of input spikes. Since such self-adaptive computing and discrimination capability on high-dimensional spatio-temporal spike patterns is not only essential for early sensory processing, but could represent a generic information processing step also in higher cortical areas, our analysis suggests to consider networks of self-organizing modules for spike-based Bayesian computation as a new model for distributed real-time information processing in the brain.

Preliminary ideas for a spike-based implementation of EM were already presented in the extended abstract [159], where we analyzed the relationship of a simple STDP rule to a Hebbian learning rule, and sketched a proof for stochastic online EM. In the present work we provide a rigorous mathematical analysis of the learning procedure, a proof of convergence, expand the framework towards learning spatio-temporal spike patterns, and discuss in detail the relationship of our STDP rule to experimental results, as well as the interpretation of spikes as samples from instantaneous posterior probability distributions in the context of EM.

4.2 Results

In this section we define a simple model circuit and show that every spiking event of the circuit can be described as one independent sample of a discrete probability distribution, which itself evolves over time in response to the spiking input. Within this network we analyze a variant of a STDP rule, in which the strength of potentiation depends on the current weight value. This local learning rule, which is supported by experimental data, and at intermediate spike frequencies closely resembles typical STDP rules from the literature, drives every synaptic weight to converge stochastically to the log of the

probability that the presynaptic input neuron fired a spike within a short time window $[t^f - \sigma, t^f]$, before the postsynaptic neuron spikes at time t^f:

$$w \to \log p(\text{presynaptic neuron fired within } [t^f - \sigma, t^f] \mid \text{postsynaptic neuron fires at } t^f) \quad . \tag{4.1}$$

We then show that the network model can be viewed as performing Bayesian computation, meaning that every spike can be understood as a sample from a posterior distribution over hidden causes in a generative probabilistic model, which combines prior probabilities and evidence from current input spike trains.

This understanding of spikes as samples of hidden causes leads to the central result of this paper. We show that STDP implements a stochastic version of Expectation Maximization for the unsupervised learning of the generative model and present convergence results for SEM. Importantly, this implementation of EM is based on spike events, rather than spike rates.

Finally we discuss how our model can be implemented with biologically realistic mechanisms. In particular this provides a link between mechanisms for lateral inhibition in WTA circuits and learning of probabilistic models. We finally demonstrate in several computer experiments that SEM can solve very demanding tasks, such as detecting and learning repeatedly occurring spike patterns, and learning models for images of handwritten digits without any supervision.

4.2.1 Definition of the network model

Our model consists of a network of spiking neurons, arranged in a WTA circuit, which is one of the most frequently studied connectivity patterns (or network motifs) of cortical microcircuits [52]. The input of the circuit is represented by the excitatory neurons $y_1, \ldots, y_n$. This input projects to a population of excitatory neurons $z_1, \ldots, z_K$ that are arranged in a WTA circuit (see Fig. 4.1). We model the effect of lateral inhibition, which is the competition mechanism of a WTA circuit [170], by a common inhibitory signal $I(t)$ that is fed to all z neurons and in turn depends on the activity of the z neurons. Evidence for such common local inhibitory signals for nearby neurons arises from numerous experimental results, see e.g. [52, 56, 61, 166]. We do not a priori impose a specific functional relationship between the common inhibition signal and the excitatory activity. Instead we will later derive necessary conditions for this relationship, and propose a mechanism that we use for the experiments.

The individual units z_k are modeled by a simplified Spike Response Model [73] in which the membrane potential is computed as the difference between the excitatory input $u_k(t)$ and the common inhibition term $I(t)$. $u_k(t)$ sums up the excitatory inputs from neurons $y_1, \ldots, y_n$ as

$$u_k(t) = w_{k0} + \sum_{i=1}^{n} w_{ki} \cdot y_i(t) \quad . \tag{4.2}$$

$w_{ki} \cdot y_i(t)$ models the EPSPs evoked by spikes of the presynaptic neuron y_i, and w_{k0} models the intrinsic excitability of the neuron z_k. In order to simplify our analysis we assume that the EPSP can be modeled as a step function with amplitude w_{ki}, i.e., $y_i(t)$ it takes on the value 1 in a finite time window of length σ after a spike and is zero before and afterwards. Further spikes within this time window do not contribute additively to the EPSP, but only extend the time window during which the EPSP is in the high state. We will later show how to extend our results to the case of realistically shaped and additive EPSPs.

We use a stochastic firing model for z_k, in which the firing probability depends exponentially on the membrane potential, i.e.,

$$p(z_k \text{ fires at time } t) \propto \exp(u_k(t) - I(t)) \quad , \tag{4.3}$$

which is in good agreement with most experimental data [113]. We can thus model the firing behavior of every neuron z_k in the WTA as an independent inhomogeneous Poisson process whose instantaneous firing rate is given by $r_k(t) = \exp(u_k(t) - I(t))$.

In order to understand how this network model generates samples from a probability distribution, we first observe that the combined firing activity of the neurons $z_1, \ldots, z_k$ in the WTA circuit is simply the sum of the K independent Poisson processes, and can thus again be modeled as an inhomogeneous Poisson process with rate $R(t) = \sum_{k=1}^{N} r_k(t)$. Furthermore, in any infinitesimally small time interval $[t, t+\delta t]$, the neuron z_k spikes with probability $r_k(t)\delta t$. Thus, if we know that at some point in time t, i.e. within $[t, t+\delta t]$, *one* of the neurons $z_1, \ldots, z_K$ produces an output spike, the conditional probability $q_k(t)$ that this spike originated from neuron z_k can be expressed as

$$q_k(t) = \frac{r_k(t)\delta t}{R(t)\delta t} = \frac{e^{u_k(t)}}{\sum_{k'=1}^{K} e^{u_{k'}(t)}} \quad . \tag{4.4}$$

Every single spike from the WTA circuit can thus be seen as an independent sample from the instantaneous distribution in Eq. (4.4) at the time of the spike. Although the instantaneous firing rate of every neuron directly depends on the value of the inhibition $I(t)$, the relative proportion of the rate $r_k(t)$ to the total WTA firing rate $R(t)$ is independent of the inhibition, because all neurons receive the same inhibition signal $I(t)$. Note that $q_k(t)$ determines only the value of the sample at time t, but not the time point at which a sample is created. The temporal structure of the sampling process depends only on the overall firing rate $R(t)$.

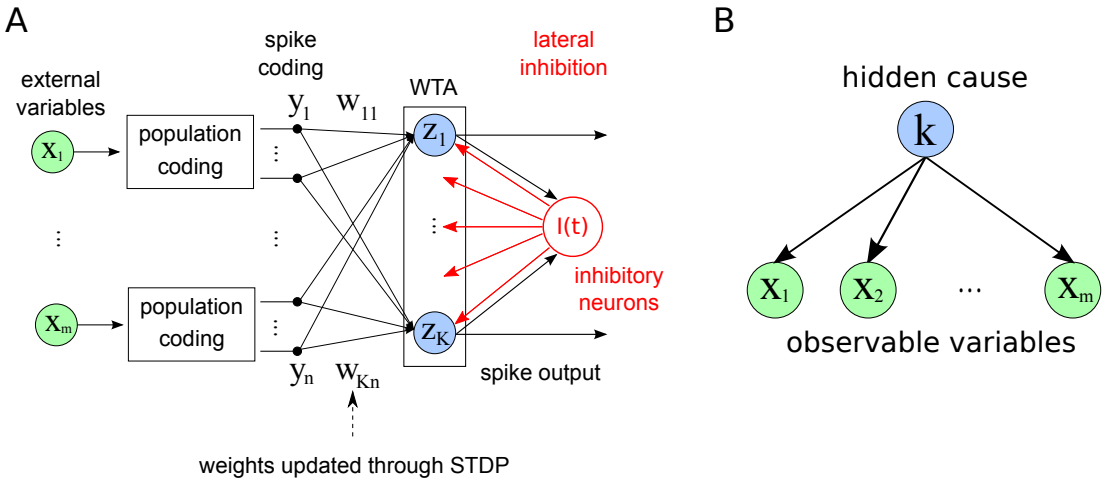

Figure 4.1: The network model and its probabilistic interpretation. A Circuit architecture. External input variables are encoded by populations of spiking neurons, which feed into a Winner-take-all (WTA) circuit. Neurons within the WTA circuit compete via lateral inhibition and have their input weights updated through STDP. Spikes from the WTA circuit constitute the output of the system. **B** Generative probabilistic model for a multinomial mixture: A vector of external input variables $x_1, \ldots, x_m$ is dependent on a hidden cause, which is represented by the discrete random variable k. In this model it is assumed that the x_i's are conditionally independent of each other, given k. The inference task is to infer the value of k, given the observations for x_i. Our neuronal network model encodes the conditional probabilities of the graphical model into the weight vector $\mathbf{w}$, such that the activity of the network can be understood as execution of this inference task.

This implementation of a stochastic WTA circuit does not constrain in any way the kind of spike patterns that can be produced. Every neuron fires independently according to a Poisson process, so it is perfectly possible (and sometimes desirable) that there are two or more neurons that fire (quasi) simultaneously. This is no contradiction to

the above theoretical argument of single spikes as samples. There we assumed that there was only one spike at a time inside a time window, but since we assumed these windows to be infinitesimally small, the probability of two spikes occurring exactly at the same point in continuous time is zero.

Synaptic and Intrinsic Plasticity

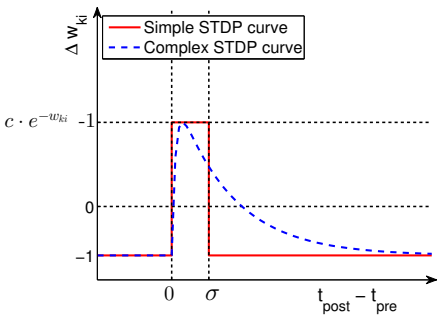

Figure 4.2: Learning curves for STDP. Under the simple STDP model (red curve), potentiation occurs only if the postsynaptic spike falls within a time window of length σ (typically 10ms) after the presynaptic spike. The convergence properties of this simpler version in conjunction with rectangular non-additive EPSPs are easier to analyze. In our simulations we use the more complex version (blue dashed curve) in combination with EPSPs that are modeled as biologically realistic α-kernels (with plausible time-constants for rise and decay of 1 respectively 15 ms).

We can now establish a link between biologically plausible forms of spike-based learning in the above network model and learning via EM in probabilistic graphical models. The synaptic weights w_{ki} of excitatory connections between input neurons y_i and neurons z_k in the WTA circuit change due to STDP. Many different versions of STDP rules have emerged from experimental data [29, 36, 207]. For synaptic connections between excitatory neurons, most of them yield a long term potentiation (LTP) when the presynaptic neuron y_i fires before the postsynaptic neuron z_k, otherwise a long term depression (LTD). In our model we use a STDP rule in which the shape of the positive update follows the shape of EPSPs at the synapses, and in which the amplitude of the update Δw_{ki} depends on the value of the synaptic weight w_{ki} before the update as in Fig. 4.2. Specifically, we propose a rule in which the ratio of LTP and LTD amplitudes is inversely exponentially dependent on the current synaptic weight. LTP curves

that mirror the EPSP shape are in accordance with previous studies, which analyzed optimal shapes of STDP curves under different mathematical criteria [174, 223]. The depression part of the rule in Fig. 4.2 is a flat offset that contrasts the potentiation. We will show later that this form of LTD occurs in our simulations only at very low repetition frequencies, and instead at natural frequencies our model gives rise to a form of STDP with spike-timing dependent LTD that is very similar to plasticity curves observed in biology [18, 207]. We will also analyze the relationship between this rule and a biologically more realistic STDP rule with an explicit time-decaying LTD part.

We can formulate this STDP-rule as a Hebbian learning rule $w_{ki} \leftarrow w_{ki} + \eta \Delta w_{ki}$ - with learning rate η - which is triggered by a spike of the postsynaptic neuron z_k at time t^f. The dependence of Δw_{ki} on the synaptic activity $y_i(t)$ and the current value of the synaptic weight is given by

$$\Delta w_{ki} = \begin{cases} ce^{-w_{ki}} - 1, & \text{if } y_i(t^f)=1, \text{ i.e. } y_i \text{ fired in } [t^f - \sigma, t^f] \\ -1, & \text{if } y_i(t^f)=0, \text{ i.e. } y_i \text{ did not fire in } [t^f - \sigma, t^f] \end{cases} . \tag{4.5}$$

Since $y_i(t)$ reflects the previously defined step function shape of the EPSP, this update rule is exactly equivalent to the simple STDP rule (solid red curve) in Fig. 4.2 for the case of the pairing of one pre- and one postsynaptic spike. The dependence on the presynaptic activity y_i is reflected directly by the time difference $t_{post} - t_{pre}$ between the pre- and the postsynaptic spikes. According to this rule positive updates are only performed if the presynaptic neuron fired in a time window of σ ms before the postsynaptic spike. This learning rule therefore respects the causality principle of LTP that is implied in Hebb's original formulation [96], rather than looking only at correlations of firing rates.

We can interpret the learning behavior of this simple STDP rule from a probabilistic perspective. Defining a stationary joint distribution $p^*(\mathbf{y}, \mathbf{z})$ over the binary input activations $\mathbf{y}$ at the times of the postsynaptic spikes, and the binary vector $\mathbf{z}$, which indicates the source of the postsynaptic spike by setting one $z_k = 1$, we show in Methods that the equilibrium condition of the expected update $\mathrm{E}[\Delta w_{ki}]$ leads to the single solution

$$\mathrm{E}[\Delta w_{ki}] = 0 \quad \Longleftrightarrow \quad w_{ki} = \log p^*(y_i{=}1 | z_k{=}1) + \log c \ . \tag{4.6}$$

This stochastic convergence to the log-probability of the presynaptic neuron being active right before the postsynaptic neuron fires is due to the exponential dependence

of the potentiation term on the current weight value. Log-probabilities are necessarily negative values, whereas for biological neural networks we typically expect excitatory, i.e. positive weights from the excitatory input neurons. The parameter c shifts the range of the values for the weights w_{ki} into the positive regime for $c > 1$. For the sake of simplicity we assume that $c = 1$ for the following theoretical analysis and we show in Methods that all results remain true for any positive value of c.

In analogy to the plasticity of the synaptic weights we also explore a form of intrinsic plasticity of the neurons. We interpret w_{k0} as an indicator for the excitability of the neuron z_k and apply a circuit-spike triggered update rule $w_{k0} \leftarrow w_{k0} + \eta \Delta w_{k0}$ with

$$\Delta w_{k0} = e^{-w_{k0}} z_k - 1 \quad . \tag{4.7}$$

Whenever a neuron z_k fires, the excitability is increased and the amount of increase is inversely exponentially dependent on the current excitability. Otherwise the excitability is decreased by a constant. Such positive feedback through use-dependent changes in the excitability of neurons were found in numerous experimental studies (see e.g. [35, 37]). This concrete model of intrinsic plasticity drives the excitability w_{k0} towards the only equilibrium point of the update rule, which is $\log p^*(z_k = 1)$. In Methods (see 'Weight offsets and positive weights') we show that the depression of the excitability can be modeled either as an effect of lateral inhibition from firing of neighboring neurons, or as a constant decay, independent of the instantaneous circuit activity. Both methods lead to different values w_{k0}, it is true, but encode identical instantaneous distributions $q_k(t)$.

Note, however, that also negative feedback effects on the excitability through homeostatic mechanisms were observed in experiments [1, 224]. In a forthcoming article [94] we show that the use of such homeostatic mechanisms instead of Eq. (4.7) in an, otherwise unchanged, network model may be interpreted as a posterior constraint in the context of EM.

Generative probabilistic model

The instantaneous spike distribution $q_k(t)$ from Eq. (4.4) can be understood as the result of Bayesian inference in an underlying generative probabilistic model for the abstract multinomial observed variables $x_1, \ldots, x_m$ and a hidden cause k. We define the probability distribution of the variables k and $\mathbf{x}$, as shown by the graphical model in

Fig. 4.1B, as $p(k, \mathbf{x}|\theta) = p(k|\theta) \cdot \prod_{j=1}^{m} p(x_j|k, \theta)$. The parametrization θ of the graphical model consists of a prior $p(k|\theta)$on k, and conditional probabilities $p(x|k, \theta)$ for every x_j.

The probabilistic model $p(k, \mathbf{x}|\theta)$ is a generative model and therefore serves two purposes: On the one hand, it can be used to generate samples of the hidden variable k and the observable variables $x_1, \ldots, x_m$. This is done by sampling k from the prior distribution, and then sampling the x_j's, which depend on k and can be generated according to the conditional probability tables. The resulting marginal distribution $p(\mathbf{x}|\theta)$ is a special case of a multinomial mixture distribution.

On the other hand, for any given observation of the vector $\mathbf{x}$, one can infer the value of the hidden cause k that led to the generation of this value for $\mathbf{x}$. By application of Bayes' rule one can infer the posterior distribution $p(k|\mathbf{x}, \theta)$ over all possible values of k, which is proportional to the product of the prior $p(k|\theta)$ and the likelihood $p(\mathbf{x}|k, \theta)$.

We define population codes to represent the external observable variables $x_1, \ldots, x_m$ by the input neurons $y_1, \ldots, y_n$, and the hidden variable k by the circuit neurons $z_1, \ldots, z_K$: For every variable x_j and every possible (discrete) value that x_j can adopt, there is exactly one neuron y_i which represents this combination. We call G_j the set of the indices of all y_i's that represent x_j, and we call $v(i)$ the possible value of x_j that is represented by neuron y_i. Thus we can define an interpretation for the spikes from the input neurons by

$$\text{neuron } y_i \text{ fires at } t^f \quad \Longrightarrow \quad x_j(t^f) = v(i) \ , \quad \text{for } i \in G_j. \tag{4.8}$$

A spike from the group G_j represents an instantaneous evidence about the observable variable x_j at the time of the spike. In the same way every neuron $z_1, \ldots, z_K$ represents one of the K possible values for the hidden variable k, and every single spike conveys an instantaneous value for k. We can safely assume that all neurons - including the input neurons - fire according to their individual local stochastic processes or at least exhibit some local stochastic jitter. For the theoretical analysis one can regard a spike as an instantaneous event at a single point in time. Thus in a continuous time no two events from such local stochastic processes can happen at exactly the same point in time. Thus, there is never more than one spike at any single point in time within a group G_j, and every spike can be treated as a proper sample from x_j. However, the neurons z_k coding for hidden causes need to integrate evidence from multiple inputs,

and thus need a mechanism to retain the instantaneous evidence from a single spike over time, in order to learn from spatial and temporal correlations in the input.

In our framework this is modeled by postsynaptic potentials on the side of the receiving neurons that are generated in response to input spikes, and, by their shape, represent evidence over time. In the simple case of the non-additive step-function model of the EPSP in Eq. (4.2), every spike indicates new evidence for the encoded variable that remains valid during a time window of σ, after which the evidence is cleared. In the case that there is no spike from one group G_j within a time window of length σ, this is interpreted as missing evidence (or missing value) for x_j in a subsequent inference. In practice it may also occur that EPSPs within a group G_j of input neurons overlap, which would indicate contradicting evidence for x_j. For the theoretical analysis we will first assume that spikes from different input neurons within the same group G_j are not closer in time than σ, in order to avoid such conflicts. We will later drop this restriction in the extension to more realistically shaped additive EPSPs by slightly enhancing the probabilistic model.

In our experiments with static input patterns we typically use the following basis scheme to encode the external input variables $x_j(t)$ by populations of stochastic spiking neurons y_i: at every point in time t there is exactly one neuron y_i in every group G_j that represents the instantaneous value of $x_j(t)$. We call this neuron the active neuron of the group, whereas all other neurons of the group are inactive. During the time where a neuron y_i is active it fires stochastically according to a Poisson processes with a certain constant or oscillating rate. The inactive neurons, however, remain silent, i.e. they fire with a rate near 0. Although not explicitly modeled here, such an effect can result from strong lateral inhibition in the input populations. This scheme certainly fulfills the definition in Eq. (4.8).

Here and in the following we will write $\mathbf{y}(t)$ to denote the input activation through the EPSPs of the network model, and $\mathbf{y}$ to denote a variable in the probabilistic model, which models the distribution of $\mathbf{y}(t)$ over all time points t. We will also use notations like $p(\mathbf{z}|\mathbf{y}(t), \mathbf{w})$, which refers to the variable $\mathbf{y}$ in the probabilistic model taking on the value $\mathbf{y}(t)$. We can then reformulate the abstract probabilistic model $p(\mathbf{x}, k|\theta)$ using the above population codes that define the binary variable vectors $\mathbf{y}$ and $\mathbf{z}$, with k s.t.

$z_k = 1$ as:

$$p(\mathbf{z}, \mathbf{y}|\mathbf{w}) = \frac{1}{Z} \sum_{k=1}^{K} z_k \cdot e^{w_{k0} + \sum_{i=1}^{n} w_{ki} \cdot y_i} \quad . \tag{4.9}$$

Under the normalization conditions

$$\sum_{k=1}^{K} e^{w_{k0}} = 1 \qquad \text{and} \qquad \forall\, k, j : \sum_{i \in G_j} e^{w_{ki}} = 1 \quad , \tag{4.10}$$

the normalization constant Z vanishes and the parametrization of the distribution simplifies to $w_{ki} = \log\ p(y_i = 1|z_k = 1, \mathbf{w})$ and $w_{k0} = \log\ p(z_k = 1|\mathbf{w})$. Even for non-normalized weights, the definition in Eq. (4.9) still represents the same type of distribution, although there is no more one-to-one mapping between the weights $\mathbf{w}$ and the parameters of the graphical model (see Methods for details). Note also that such log-probabilities are exactly (up to additive constants) the local equilibrium points in Eq. (4.6) of the STDP rule in Fig. 4.2. In the section "STDP approximates Expectation Maximization" we will discuss in detail how this leads to unsupervised learning of a generative model of the input data in a WTA circuit.

Spike-based Bayesian computation

We can now formulate an exact link between the above generative probabilistic model and our neural network model of a simplified spike-based WTA circuit. We show that at any point in time t^f at which the network generates an output spike, the relative firing probabilities $q_k(t^f)$ of the output neurons z_k as in Eq. (4.4), are equal to the posterior distribution of the hidden cause k, given the current evidences encoded in the input activations $\mathbf{y}(t^f)$. For a given input $\mathbf{y}(t^f)$ we use Bayes' rule to calculate the posterior probability of cause k as $p(k|\mathbf{y}(t^f), \mathbf{w})$. We can identify the prior $p(k|\mathbf{w})$ with the excitabilities w_{k0} of the neurons. The log-likelihood $\log p(\mathbf{y}(t^f)|k, \mathbf{w})$ of the current evidences given the cause k corresponds to the sum of excitatory EPSPs, which depend on the synaptic weights w_{ki}. This leads to the calculation

$$p(k|\mathbf{y}(t^f),\mathbf{w}) = \frac{\overbrace{e^{w_{k0}}}^{\text{prior } p(k|\mathbf{w})} \cdot \overbrace{e^{\sum w_{ki}y_i(t^f)}}^{\text{likelihood } p(\mathbf{y}(t^f)|k,\mathbf{w})}}{\underbrace{\sum_{k'=1}^{K} e^{w_{k'0}+\sum w_{k'i}y_i(t^f)}}_{p(\mathbf{y}(t^f)|\mathbf{w})}} = \frac{e^{u_k(t^f)}}{\sum_{k'=1}^{K} e^{u_{k'}(t^f)}} = q_k(t^f) \quad . \quad (4.11)$$

This shows that at all times t^f every spike from the WTA circuit represents one sample of the instantaneous posterior distribution $p(k|\mathbf{y}(t^f),\mathbf{w})$.

The crucial observation, however, is that this relation is valid at any point in time, independently of the inhibitory signal $I(t)$. It is only the ratio between the quantities $e^{u_k(t)}$ that determines the relative firing probabilities $q_k(t)$ of the neurons z_k.

Background oscillations and learning with missing values

We will now show that for the case of a low average input firing rate, a modulation of the firing rate can be beneficial, as it can synchronize firing of pre- and post-synaptic neurons. Each active neuron then fires according to an inhomogeneous Poisson process, and we assume for simplicity that the time course of the spike rate for all neurons follows the same oscillatory (sinusoidal) pattern around a common average firing rate. Nevertheless the spikes for each y_i are drawn as samples from independent processes. In addition, let the common inhibition signal $I(t)$ be modulated by an additional oscillatory current $I_{osc}(t) = A \cdot \sin(\omega t + \phi)$ with amplitude A, oscillation frequency ω (same as for the input oscillation), and phase shift ϕ. Due to the increased number of input neurons firing simultaneously, and the additional background current, pre- and post-synaptic firing of active neurons will synchronize. The frequency of the background oscillation can be chosen in principle arbitrarily, as long as the number of periods per input example is constant. Otherwise the network will weight different input examples by the number of peaks during presentation, which might lead to learning of a different generative model.

The effect of a synchronization of pre- and post-synaptic firing can be very beneficial, since at low input firing rates it might happen that none of the input neurons in a population of neurons encoding an external variable x_j fires within the integration time window of length σ of output neurons z_k. This corresponds to learning with missing

attribute values for x_j, which is known to impair learning performance in graphical models [74]. Our novel interpretation is therefore that background oscillations can reduce the percentage of missing values by synchronizing presynaptic firing rates. This agrees with previous studies, which have shown that it is easier for single detector neurons learning with phenomenological STDP rules to detect spike patterns embedded in a high-dimensional input stream, if the patterns are encoded relative to a background oscillation [149], or the patterns consist of dense and narrow bursts of synchronous activity [77]. These results still hold if only a small part of the afferents participates in the pattern, or spikes from the pattern are missing, since the increased synchrony facilitates the identification of the pattern. Although we show in experiments that this increased synchronization can improve the learning performance of spike-based probabilistic learners in practice, it is important to note that background oscillations are not necessary for the theory of spike-based Expectation Maximization to hold. Also, brain oscillations have previously been associated with various fundamental cognitive functions like e.g. attention, memory, consciousness, or neural binding. In contrast, our suggested role for oscillations as a mechanism for improving learning and inference with missing values is very specific within our framework, and although some aspects are compatible with higher-level theories, we do not attempt here to provide alternative explanations for these phenomena.

Our particular model of oscillatory input firing rates leaves the average firing rates unchanged, hence the effect of oscillations does not simply arise due to a larger number of input or output spikes. It is the increased synchrony of input and output spikes by which background oscillations can facilitate learning for tasks in which inputs have little redundancy, and missing values during learning thus would have a strong impact. We demonstrate this in the following experiment, where a common background oscillation for the input neurons y_i and the output neurons z_k significantly speeds up and improves the learning performance. In other naturally occurring input distributions with more structured inputs, oscillations might not improve the performance.

4.2.2 Example 1: Learning of probabilistic models with STDP

Fig. 4.3 demonstrates the emergence of Bayesian computation in the generic network motif of Fig. 4.1A in a simple example. Spike inputs $\mathbf{y}$ (top row of Fig. 4.3D) are generated through four different hidden processes (associated with four different colors).

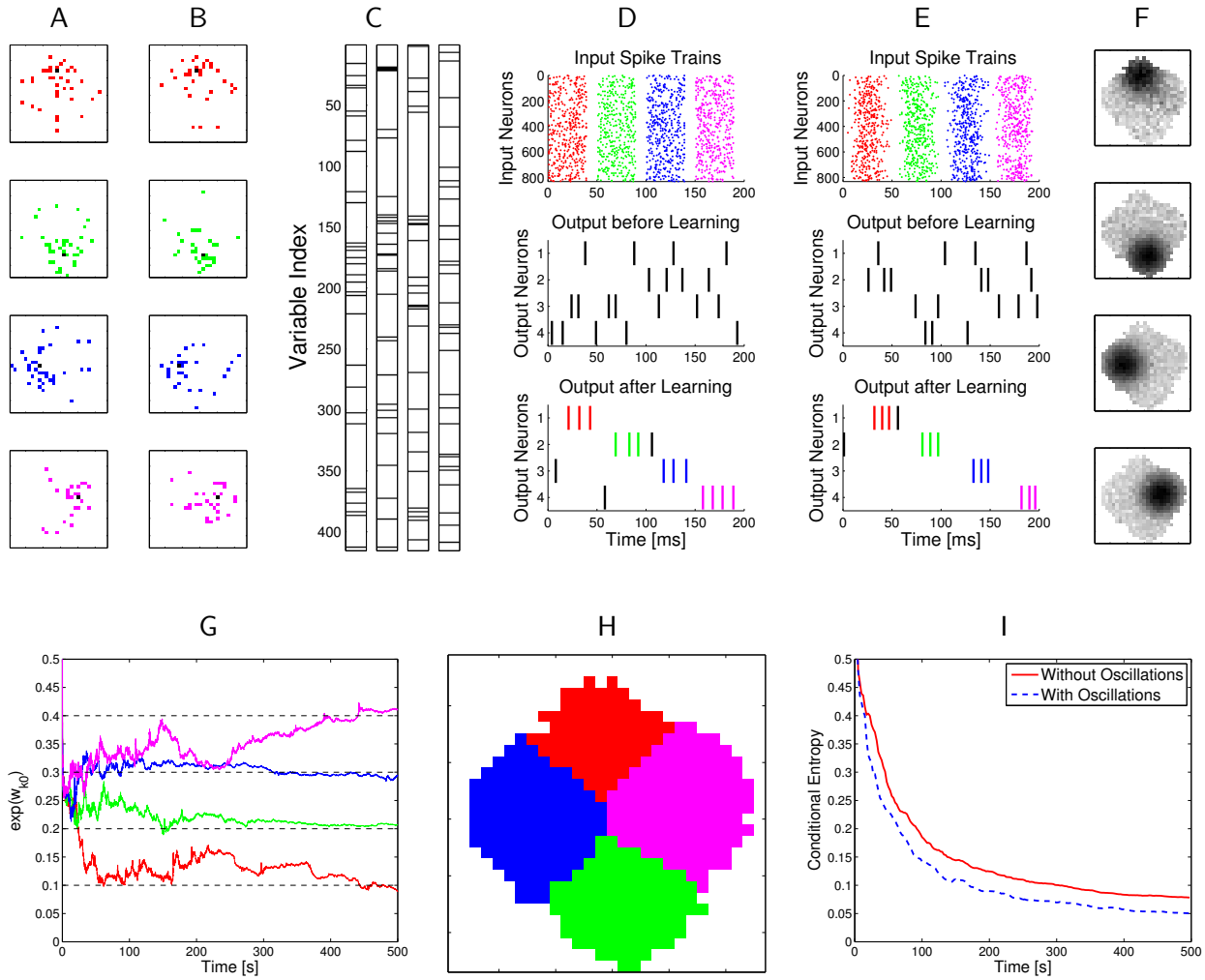

Figure 4.3: Example for the emergence of Bayesian computation through STDP and adaptation of neural excitability. A, B: Visualization of hidden structurin the spike inputs **y** shown in D, E: Each row in panels A and B shows two results of drawing pixels from the same Gauss distribution over a 28 x 28 pixel array. Four different Gauss distributions were used in the four rows, and the location of their center represents the latent variable behind the structure of the input spike train. **C**: Transformation of the four 2D images in B into four linear arrays, resulting from random projections from 2D locations to 1D indices. Black lines indicate active pixels, and pixels that were active in less than 4 % of all images were removed before the transformation (these pixels are white in panel H). By the random projection, both the 2D structure of the underlying pixel array and the value of the latent variable are hidden when the binary 1D vector is encoded through population coding into the spike trains **y** that the neural circuit receives. **D**: Top row: Spike trains from 832 input neurons that result from the four linear patterns shown in panel C (color of spikes indicates which of the four hidden processes had generated the underlying 2D pattern, after 50 ms another 2D pattern is encoded). *Continued on next page...*

Continued: Caption for Figure 4.3. The middle and bottom row show the spike output of the four output neurons at the beginning and after 500 s of unsupervised learning with continuous spike inputs (every 50 ms another 2D pattern was randomly drawn from one of the 4 different Gauss distributions, with different prior probabilities of 0.1, 0.2, 0.3, and 0.4.). Color of spikes indicates the emergent specialization of the four output neurons on the four hidden processes for input generation. Black spikes indicate incorrect guesses of hidden cause. **E**: Same as D, but with a superimposed 20 Hz oscillation on the firing rates of input neurons and membrane potentials of the output neurons. Fewer error spikes occur in the output, and output spikes are more precisely timed. **F**: Internal models (weight vectors $\mathbf{w}$) of output neurons $z_1, \ldots, z_4$ after learning (pixel array). **G**: Autonomous learning of priors $p(k) \approx e^{w_{k0}}$, that takes place simultaneously with the learning of internal models. **H**: Average "winner" among the four output neurons for a test example (generated with equal probability by any of the 4 Gaussians) when a particular pixel was drawn in this test example, indicating the impact of the learned priors on the output response. **I**: Emergent discrimination capability of the output neurons during learning (red curve). The dashed blue curve shows that a background oscillation as in E speeds up discrimination learning. Curves in G and I represent averages over 20 repetitions of the learning experiment.

Each of them is defined by a Gauss distribution over a 2D pixel array with a different center, which defines the probability of every pixel to be on. Spike trains encode the current value of a pixel by a firing rate of 25 Hz or 0 Hz for 40 ms. Each pixel was encoded by two input neurons y_i via population coding, exactly one of them had a firing rate of 25 Hz for each input image. A 10 ms period without firing separates two images in order to avoid overlap of EPSPs for input spikes belonging to different input images.

After unsupervised learning with STDP for 500 s (applied to continuous streams of spikes as in panel D of Fig. 4.3) the weight vectors shown in Fig. 4.3F (projected back into the virtual 2D input space) emerged for the four output neurons z_1, z_2, z_3, z_4, demonstrating that these neurons had acquired internal models for the four different processes that were used to generate inputs. The four different processes for generating the underlying 2D input patterns had been used with different prior probabilities $(0.1, 0.2, 0.3, 0.4)$. Fig. 4.3G shows that this imbalance resulted in four different priors $p(k)$ encoded in the biases $e^{w_{k0}}$ of the neurons z_k. When one compares the unequal sizes of the colored areas in Fig. 4.3H with the completely symmetric internal models (or likelihoods) of the four neurons shown in panel F, one sees that their firing probability approximates a posterior over hidden causes that results from multiplying their

learned likelihoods with their learned priors. As a result, the spike output becomes sparser, and almost all neurons only fire when the current input spikes are generated by that one of the four hidden processes on which they have specialized (Fig. 4.3D, bottom row). In Fig. 4.3I the performance of the network is quantified over time by the normalized conditional entropy $H(k|\zeta_{out})/H(k,\zeta_{out})$, where k is the correct hidden cause of each input image $\mathbf{y}$ in the training set, and ζ_{out} denotes the discrete random variable defined by the firing probabilities of output neurons z_k for each image under the currently learned model. Low conditional entropy indicates that each neuron learns to fire predominantly for inputs from one class. Fig. 4.3E as well as the dashed blue line in Fig. 4.3I show that the learning process is improved when a common background oscillation at 20 Hz is superimposed on the firing rate of input neurons and the membrane potential of the output neurons, while keeping the average input and output firing rates constant. The reason is that in general it may occur that an output neuron z_k receives during its integration time window (40 ms in this example) no information about the value of a pixel (because neither the neuron y_i that has a high firing rate for 40 ms if this pixel is black, nor the associated neuron $y_{i'}$ that has a high firing rate if this pixel is white fire during this time window). A background oscillation reduces the percentage of such missing values by driving presynaptic firing times together (see top row of Fig. 4.3E). Note that through these oscillations the overall output firing rate $R(t)$ fluctuates strongly, but since the same oscillation is used consistently for all four types of patterns, the circuit still learns the correct distribution of inputs.

This task had been chosen to become very fast unsolvable if many pixel values are missing. Many naturally occurring input distributions, like the ones addressed in the subsequent computer experiments, tend to have more redundancy, and background oscillations did not improve the learning performance for those.

4.2.3 STDP approximates Expectation Maximization

In this section we will develop the link between the unsupervised learning of the generative probabilistic model in Fig. 4.1B and the learning effect of STDP as defined in our spiking network model in Fig. 4.1A. Starting from a learning framework derived from the concept of Expectation Maximization [44], we show that the biologically plausible STDP rule from Fig. 4.2 can naturally approximate a stochastic, online version of this optimization algorithm. We call this principle SEM (spike-based EM).

SEM can be viewed as a bootstrapping procedure. The relation between the firing probabilities of the neurons within the WTA circuit and the continuous updates of the synaptic weights with our STDP rule in Eq. (4.5) drive the initially random firing of the circuit in response to an input $\mathbf{y}$ towards learning the correct generative model of the input distribution. Whenever a neuron z_k fires in response to $\mathbf{y}$, the STDP rule increases the weights w_{ki} of synapses from those presynaptic neurons y_i that had fired shortly before z_k. In absence of a recent presynaptic spike from y_i the weight w_{ki} is decreased. As a consequence, when next a pattern similar to $\mathbf{y}$ is presented, the probability for the same z_k to fire and further adapt its weights, is increased. Since z_k becomes more of an "expert" for one subclass of input patterns, it actually becomes less likely to fire for non-matching patterns. The competition in the WTA circuit ensures that other z-neurons learn to specialize for these different input categories.

In the framework of Expectation Maximization, the generation of a spike in a z-neuron creates a sample from the currently encoded posterior distribution of hidden variables, and can therefore be viewed as the stochastic Expectation, or E-step. The subsequent application of STDP to the synapses of this neuron can be understood as an approximation of the Maximization, or M-step. The online learning behavior of the network can be understood as a stochastic online EM algorithm.

Learning the parameters of the probability model by EM

The goal of learning the parametrized generative probabilistic model $p(\mathbf{y}, k|\mathbf{w})$ is to find parameter values $\mathbf{w}$, such that the marginal distribution $p(\mathbf{y}|\mathbf{w})$ of the model distribution approximates the actual stationary distribution of spike inputs $p^*(\mathbf{y})$ as closely as possible. We define $p^*(\mathbf{y})$ as the probability to observe the activation vector $\mathbf{y}(t)$ at some point t in time (see Eq. (4.72) in Methods for a precise mathematical definition). The learning task can thus be formalized as the minimization of the Kullback-Leibler divergence between the two distributions, $p(\mathbf{y}|\mathbf{w})$ and $p^*(\mathbf{y})$. A mathematically equivalent formulation is the maximization of the expected likelihood $L(\mathbf{w}) = E_{p^*}[\log p(\mathbf{y}|\mathbf{w})]$ of the inputs $\mathbf{y}$, drawn from $p^*(\mathbf{y})$. The parametrization of the generative probabilistic model $p(\mathbf{y}, k|\mathbf{w})$ is highly redundant, i.e. for every $\mathbf{w}$ there is a continuous manifold of $\mathbf{w}'$, that all define identical generative distributions $p(\mathbf{y}, k|\mathbf{w}')$ in Eq. (4.24). There is, however, exactly one $\mathbf{w}'$ in this sub-manifold of the weight space that fulfills the normalization conditions in Eq. (4.10). By imposing the normalization conditions as

constraints to the maximization problem, we can thus find unique local maxima (see "Details to Learning the parameters of the probability model by EM" in Methods).

The most common way to solve such unsupervised learning problems with hidden variables is the mathematical framework of Expectation Maximization (EM). In its standard form, the EM algorithm is a batch learning mechanism, in which a fixed, finite set of T instances of input vectors $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(T)}$ is given, and the task is to find the parameter vector $\mathbf{w}$ that maximizes the log-likelihood $L(\mathbf{w}) = \sum_{l=1}^{T} \log p(\mathbf{y}^{(l)}|\mathbf{w})$ of these T instances to be generated as independent samples by the model $p(\mathbf{y}|\mathbf{w})$.

Starting from a random initialization for $\mathbf{w}$, the algorithm iterates between E-steps and M-steps. In the E-steps, the current parameter vector $\mathbf{w}$ is used to find the posterior distributions of the latent variables $k^{(1)}, \ldots, k^{(T)}$, each given by $p(k^{(l)}|\mathbf{y}^{(l)}, \mathbf{w})$.

In the M-steps a new parameter vector $\mathbf{w}^{\text{new}}$ is computed, which maximizes the expected value of the complete-data log-likelihood function, subject to the normalization constraints in Eq. (4.10). The analytical solution for this M-step (compare [22]) is given by

$$w_{ki}^{\text{new}} := \log \frac{\sum_{l=1}^{T} y_i^{(l)} p(k|\mathbf{y}^{(l)}, \mathbf{w})}{\sum_{l=1}^{T} p(k|\mathbf{y}^{(l)}, \mathbf{w})} \quad \text{and} \quad w_{k0}^{\text{new}} := \log \frac{\sum_{l=1}^{T} p(k|\mathbf{y}^{(l)}, \mathbf{w})}{T} \quad . \tag{4.12}$$

The iterated application of this update procedure is guaranteed to converge to a (local) maximum of $L(\mathbf{w})$ [44]. It is obvious that $\mathbf{w}^{\text{new}}$ fulfills the desired normalization conditions in Eq. (4.10) after every update.

Although the above deterministic algorithm requires that the same set of T training examples is re-used for every EM iteration, similar results also hold valid for online learning scenarios. In an online setup new samples $\mathbf{y}(l) \propto p^*(\mathbf{y})$ are drawn from the input distribution at every iteration, which is closer to realistic neural network learning settings. Instead of analytically computing the expected value of the complete-data log-likelihood function, a Monte-Carlo estimate is computed using the samples $k^{(l)}$, drawn according to their posterior distribution $p(k|\mathbf{y}^{(l)}, \mathbf{w})$. Even though additional stochastic fluctuations are introduced due to the stochastic sampling process, this stochastic EM algorithm will also converge to a stable result in the limit of infinite iterations, if the number of samples T is increased with every iteration [110].

In order to simplify the further notation we introduce the augmented input distribution $p^*_{\mathbf{w}}(\mathbf{y}, \mathbf{z})$ from which we can sample pairs $\langle \mathbf{y}, \mathbf{z} \rangle$ and define

$$p^*_{\mathbf{w}}(\mathbf{y}, \mathbf{z}) = p(\mathbf{z}|\mathbf{y}, \mathbf{w}) p^*(\mathbf{y}) \quad . \tag{4.13}$$

Sampling pairs $\langle \mathbf{y}^{(l)}, \mathbf{z}^{(l)} \rangle$ with $l = 1, \ldots, T$ from $p^*_{\mathbf{w}}(\mathbf{y}, \mathbf{z})$ corresponds to online sampling of inputs, combined with a stochastic E-step. The subsequent M-step

$$w^{\text{new}}_{ki} := \log \frac{\sum_{l=1}^{T} y_i^{(l)} z_k^{(l)}}{\sum_{l=1}^{T} z_k^{(l)}} \quad , \quad w^{\text{new}}_{k0} := \log \frac{\sum_{l=1}^{T} z_k^{(l)}}{T} \tag{4.14}$$

essentially computes averages over all T samples: $\exp(w^{\text{new}}_{k0})$ is the average of the variable z_k; $\exp(w^{\text{new}}_{ki})$ is a conditional average of y_i taken over those instances in which z_k is 1.

The expected value of the new weight vector after one iteration, i.e., the sampling E-step and the averaging M-step, can be expressed in a very compact form based on the augmented input distribution as

$$\mathrm{E}_{p^*_{\mathbf{w}}}[w^{\text{new}}_{ki}] = \log p^*_{\mathbf{w}}(y_i = 1 | z_k = 1) \qquad \mathrm{E}_{p^*_{\mathbf{w}}}[w^{\text{new}}_{k0}] = \log p^*_{\mathbf{w}}(z_k = 1) \quad . \tag{4.15}$$

A necessary condition for a point convergence of the iterative algorithm is a stable equilibrium point, i.e. a value $\mathbf{w}$ at which the expectation of the next update $\mathbf{w}^{\text{new}}$ is identical to $\mathbf{w}$. Thus we arrive at the following necessary implicit condition for potential convergence points of this stochastic algorithm.

$$w_{ki} = \log p^*_{\mathbf{w}}(y_i = 1 | z_k = 1) \qquad w_{k0} = \log p^*_{\mathbf{w}}(z_k = 1) \quad . \tag{4.16}$$

This very intuitive implicit "solution" is the motivation for relating the function of the simple STDP learning rule (solid red line in Fig. 4.2) in the neural circuit shown in Fig. 4.1A to the framework of EM.

Spike-based Expectation Maximization

In order to establish a mathematically rigorous link between the STDP rule in Fig. 4.2 in the spike-based WTA circuit and stochastic online EM we identify the functionality of both the E- and the M-steps with the learning behavior of the spiking WTA-circuit with STDP.

In a biologically plausible neural network setup, one cannot assume that observations are stored and computations necessary for learning are deferred until a suitable sample size has been reached. Instead, we relate STDP learning to online learning algorithms in the spirit of Robbins-Monro stochastic approximations, in which updates are performed after every observed input.

At an arbitrary point in time t^f at which any one neuron z_k of the WTA circuit fires, the posterior $p(k|\mathbf{y}(t), \mathbf{w})$ according to Eq. (4.4) gives the probability that the spike at this time t^f has originated from the neuron with index k. The pair $\langle \mathbf{y}(t), k \rangle$ can therefore be seen as a sample from the augmented input distribution $p^*_{\mathbf{w}}(\mathbf{y}, k)$. Hence, we can conclude that the generation of a spike by the WTA circuit corresponds to the generation of samples $\langle \mathbf{y}, k \rangle$ during the E-step. There are additional conditions on the inhibition signal $I(t)$ that have to be met in order to generate unbiased samples $\mathbf{y}(t^f)$ from the input distribution $p^*(\mathbf{y})$. These are discussed in depth in the section "Role of the Inhibition", but for now let us assume that these conditions are fulfilled.

The generation of a spike in the postsynaptic neuron z_k triggers an STDP update according to Eq. (4.5) in all synapses from incoming presynaptic neurons y_i, represented by weights w_{ki}. We next show that the biologically plausible STDP rule in Eq. (4.5) (see also Fig. 4.2) together with the rule in Eq. (4.7) can be derived as approximating the M-step in stochastic online EM.

The update in Eq. (4.14) suggests that every synapse w_{ki} collects the activation statistics of its input y_i (the presynaptic neuron), given that its output z_k (the postsynaptic neuron) fires. These statistics can be gathered online from samples of the augmented input distribution $p^*_{\mathbf{w}}(\mathbf{y}, \mathbf{z})$.

From this statistical perspective each weight can be interpreted as $w_{ki} = \log \frac{a_{ki}}{N_{ki}}$, where a_{ki} and N_{ki} are two local virtual counters in each synapse. a_{ki} represents the number of the events $\langle y_i = 1, z_k = 1 \rangle$ and N_{ki} represents the number of the events $\langle z_k = 1 \rangle$, i.e. the postsynaptic spikes. Even though all virtual counters N_{ki} within one neuron z_k count the same postsynaptic spikes, it is easier to think of one individual such counter for every synapse. If we interpret the factor $\frac{1}{N_{ki}}$ as a local learning rate η_{ki}, we can derive Eq. (4.5) (see Methods) as the spike-event triggered stochastic online learning rule $w^{\text{new}}_{ki} = w_{ki} + \eta_{ki} z_k (y_i e^{-w_{ki}} - 1)$ that approximates in the synapse w_{ki} the log of the running average of $y_i(t^f)$ at the spiking times of neuron z_k. The update formula shows that w_{ki} is only changed, if the postsynaptic neuron z_k fires, whereas

spike events of other neurons $\langle z_{k'} = 1 \rangle$ with $k' \neq k$ are irrelevant for the statistics of w_{ki}. Thus the learning rule is purely local for every synapse w_{ki}; it only has to observe its own pre- and postsynaptic signals. Additionally we show in the Methods section "Adaptive learning rates with Variance tracking" a very efficient heuristic how the learning rate eta_{ki} can be estimated locally.

Analogously we can derive the working mechanism of the update rule in Eq. (4.7) as updates of the log of a fraction at the respective points in time.

The simple STDP rules in Eq. (4.5) and Eq. (4.7) thus approximate the M-step in a formal generative probabilistic model with local, biologically plausible computations. It remains to be shown that these STDP rules actually drive the weights $\mathbf{w}$ to converge to the target points in Eq. (4.16) of the stochastic EM algorithm.

We can conclude from the equilibrium conditions of the STDP rule in Eq. (4.6) that convergence can only occur at the desired local maxima of the likelihood $L(\mathbf{w})$ subject to the normalization constraints. However, it remains to be shown that the update algorithm converges at all and that there are no limit cycles.

Proof of convergence

Even though we successfully identified the learning behavior of the simple STDP rule (Fig. 4.2) in the circuit model with the E- and the M-steps of the EM algorithm, this is not yet sufficient for a complete proof of convergence for the whole learning system. Not only are the single updates just approximations to the M-step, these approximations, in addition, violate the normalization conditions in Eq. (4.10). Although the system - as we will show - converges towards normalized solutions, there is always a stochastic fluctuation around the normalization conditions. One can therefore not simply argue that Eq. (4.5) implements a stochastic version of the generalized EM algorithm; instead, we have to resort to the theory of stochastic approximation algorithms as presented in [131]. Under some technical assumptions (see Methods) we can state

Theorem 1: *The algorithm in Eq. (4.5,4.7) updates* $\mathbf{w}$ *in a way that it converges with probability 1 to the set of local maxima of the likelihood function* $L(\mathbf{w}) = E_{p^*}[\log p(\mathbf{y}|\mathbf{w})]$*, subject to the normalization constraints in Eq. (4.10).*

The detailed proof, which is presented in Methods, shows that the expected trajectory of the weight vector $\mathbf{w}$ is determined by two driving forces. The first one is a normalization force which drives $\mathbf{w}$ from every arbitrary point towards the regime

where $\mathbf{w}$ is normalized. The second force is the real learning force that drives $\mathbf{w}$ to a desired maximum of $L(\mathbf{w})$. However, this interpretation of the learning force is valid only if $\mathbf{w}$ is sufficiently close to normalized.

4.2.4 The Role of the Inhibition

We have previously shown that the output spikes of the WTA circuit represent samples from the posterior distribution in Eq. (4.11), which only depends on the ratios between the membrane potentials $u_k(t)$. The rate at which these samples are produced is the overall firing rate $R(t)$ of the WTA circuit and can be controlled by modifying the common inhibition $I(t)$ of the neurons z_k.

Although any time-varying output firing rate $R(t)$ produces correct samples from the posterior distribution in Eq. (4.11) of $\mathbf{z}$, for learning we also require that the input patterns $\mathbf{y}(t)$ observed at the spike times are unbiased samples from the true input distribution $p^*(\mathbf{y})$. If this is violated, some patterns coincide with a higher $R(t)$, and thus have a stronger influence on the learned synaptic weights. In Methods we formally show that $R(t)$ acts as a multiplicative weighting of the current input $\tilde{\mathbf{y}}(t)$, and so the generative model will learn a slightly distorted input distribution.

An unbiased set of samples can be obtained if $R(t)$ is independent of the current input activation $\mathbf{y}(t)$, e.g. if $R(t) = R$ is constant. This could in theory be achieved if we let $I(t)$ depend on the current values of the membrane potentials $u_k(t)$, and set $I(t) = -\log R + \log \sum_{k=1}^{K} e^{u_k(t)}$. Such an immediate inhibition is commonly assumed in rate-based soft-WTA models, but it seems implausible to compute this in a spiking neuronal network, where only spikes can be observed, but not the presynaptic membrane potentials.

However, our results show that a perfectly constant firing rate is not a prerequisite for convergence to the right probabilistic model. Indeed we can show that it is sufficient that $R(t)$ and $\mathbf{y}(t)$ are stochastically independent, i.e. $R(t)$ is not correlated to the appearance of any specific value of $\mathbf{y}(t)$. Still this might be difficult to achieve since the firing rate $R(t)$ is functionally linked to the input $\mathbf{y}(t)$ by $R(t) = e^{-I(t)}\, Z\, p(\mathbf{y}(t)|\mathbf{w})$, but it clarifies the role of the inhibition $I(t)$ as de-correlating $R(t)$ from the input $\mathbf{y}$, at least in the long run.

One possible biologically plausible mechanism for such a decorrelation of $R(t)$ and $\mathbf{y}(t)$ is an inhibitory feedback from a population of neurons that is itself excited by

the neurons z_k. Such WTA competition through lateral inhibition has been studied extensively in the literature [52, 170]. In the implementation used for the experiments in this paper every spike from the z-neurons causes an immediate very strong inhibition signal that lasts longer than the refractory period of the spiking neuron. This strong inhibition decays exponentially and is overlaid by a noise signal with high variability that follows an Ornstein-Uhlenbeck process (see "Inhibition Model in Computer Simulations" in Methods). This will render the time of the next spike of the system almost independent of the value of $p(\mathbf{y}(t)|\mathbf{w})$.

It should also be mentioned that a slight correlation between $R(t)$ and $p(\mathbf{y}(t)|\mathbf{w})$ may be desirable, and $I(t)$ might also be externally modulated (for example through attention, or neuromodulators such as Acetylcholin), as an instrument of selective input learning. This might lead e.g. to slightly higher firing rates for well-known inputs (high $p(\mathbf{y}(t)|\mathbf{w})$), or salient inputs, as opposed to reduced rates for unknown arbitrary inputs. In general, however, combining online learning with a sampling rate $R(t)$ that is correlated to $p(\mathbf{y}|\mathbf{w})$ may lead to strange artifacts and might even prohibit the convergence of the system due to positive feedback effects. A thorough analysis of such effects and of possible learning mechanisms that cope with positive feedback effects is the topic of future research.

Our theoretical analysis sheds new light on the requirements for inhibition in spiking WTA-like circuits to support learning and Bayesian computation. Inhibition does not only cause competition between the excitatory neurons, but also regulates the overall firing rate $R(t)$ of the WTA circuit. Variability in $R(t)$ does not influence the performance of the circuit, as long as there is no systematic dependence between the input and $R(t)$.

4.2.5 Continuous-Time Interpretation with Realistically Shaped EPSPs

In our previous analysis we have assumed a simplified non-additive step-function model for the EPSP. This allowed us to describe all input evidence within the last time window of length σ by one binary vector $\mathbf{y}(t)$, but required us to assume that no two neurons within the same group G_j fired within that period. We will now give an intuitive explanation to show that this restriction can be dropped and present an interpretation for additive biologically plausibly shaped EPSPs as inference in a generative model.

The postsynaptic activation $\tilde{y}_i(t)$ under an additive EPSPs is given by the convolution

$$\tilde{y}_i(t) = \sum_f K(t - t_i^f) \quad , \tag{4.17}$$

where K describes an arbitrarily shaped kernel, e.g. an α-shaped EPSP function which is the difference of two exponential functions (see [73]) with different time constants. We use 1 ms for the rise and 15 ms for the decay in our simulations. $\tilde{y}_i(t)$ replaces $y_i(t)$ in Eq. (4.2) in the computation of the membrane potential $u_k(t)$ of our model neurons. We can still understand the firing of neurons in the WTA circuit according to the relative firing probabilities $q_k(t)$ in Eq. (4.4) as Bayesian inference. To see this, we imagine an extension of the generative probabilistic model $p(\mathbf{x}, k|\theta)$ in Fig. 4.1B, which contains multiple instances of $\mathbf{x}$, exactly one for every input spike from all input neurons y_i. For a fixed common hidden cause k, all instances of $\mathbf{x}$ are conditionally independent of each other, and have the same conditional distributions for each x_j (see Methods for the full derivation of the extended probabilistic model). According to the definition in Eq. (4.8) of the population code every input spike represents evidence that x_j in an instance $\mathbf{x}$ should take on a certain value. Since every spike contributes only to one instance, any finite input spike pattern can be interpreted as valid evidence for multiple instances of inputs $\mathbf{x}$.

The inference of a single hidden cause k in such extended graphical model from multiple instances of evidence is relatively straightforward: due to the conditional independence of different instances, we can compute the input likelihood for any hidden cause simply as the product of likelihoods for every single evidence. Inference thus reduces to counting how often every possible evidence occurred in all instances $\mathbf{x}$, which means counting the number of spikes of every y_i. Since single likelihoods are implicitly encoded in the synaptic weights w_{ki} by the relationship $w_{ki} = \log p(y_i = 1|k, \mathbf{w})$, we can thus compute the complete input likelihood by adding up step-function like EPSPs with amplitudes corresponding to w_{ki}. This yields correct results, even if one input neuron spikes multiple times.

In the above model, the timing of spikes does not play a role. If we want to assign more weight to recent evidence, we can define a heuristic modification of the extended graphical model, in which contributions from spikes to the complete input log-likelihood

are linearly interpolated in time, and multiple pieces of evidence simply accumulate. This is exactly what is computed in $\tilde{y}_i$ in Eq. (4.17), where the shape of the kernel $K(t - t^f)$ defines how the contribution of an input spike at time t^f evolves over time. Defining $\tilde{y}_i$ as the weight for the evidence of the assignment of x_j to value $v(i)$, it is easy to see (and shown in detail in Methods) that the instantaneous output distribution $q_k(t)$ represents the result of inference over causes k, given the time-weighted evidences of all previous input spikes, where the weighting is done by the EPSP-function $K(t)$. Note that this evidence weighting mechanism is not equivalent to the much more complex mechanism for inference in presence of uncertain evidence, which would require more elaborate architectures than our feed-forward WTA-circuit. In our case, past evidence does not become uncertain, but just less important for the inference of the instantaneous hidden cause k.

We can analogously generalize the spike-triggered learning rule in Eq. (4.5) for continuous-valued input activations $\tilde{y}_i(t)$ according to Eq. (4.17):

$$\Delta w_{ki}(t) = \tilde{y}_i(t) \cdot c \cdot e^{-w_{ki}} - 1 \quad . \tag{4.18}$$

The update of every weight w_{ki} is triggered when neuron z_k, i.e. the postsynaptic neuron, fires a spike. The shape of the LTP part of the STDP curve is determined by the shape of the EPSP, defined by the kernel function $K(t)$. The positive part of the update in Eq. (4.18) is weighted by the value of $\tilde{y}_i(t)$ at the time of firing the postsynaptic spike. Negative updates are performed if $\tilde{y}_i(t)$ is close to zero, which indicates that no presynaptic spikes were observed recently. The complex version of the STDP curve (blue dashed curve in Fig. 4.1B), which resembles more closely to the experimentally found STDP curves, results from the use of biologically plausible α-shaped EPSPs. In this case, the LTP window of the weight update decays with time, following the shape of the α-function. This form of synaptic plasticity was used in all our experiments. If EPSPs accumulate due to high input stimulation frequencies, the resulting shape of the STDP curve becomes even more similar to previously observed experimental data, which is investigated in detail in the following section.

The question remains, how this extension of the model and the heuristics for time-dependent weighting of spike contributions affect the previously derived theoretical properties. Although the convergence proof does not hold anymore under such general conditions we can expect (and show in our Experiments) that the network will still show

the principal behavior of EM under fairly general assumptions on the input: we have to assume that the instantaneous spike rate of every input group G_j is not dependent on the value of x_j that it currently encodes, which means that the total input spike rate must not depend on the hidden cause k. Note that this assumption on every input group is identical to the desired output behavior of the WTA circuit according to the conditions on the inhibition as derived earlier. This opens up the possibility of building networks of recursively or hierarchically connected WTA circuits. Note also that the grouping of inputs into different G_j is only a notational convenience. The neurons in the WTA circuit do not have to know which inputs are from the same group, neither for inference nor for learning, and can thus treat all input neurons equally.

4.2.6 Relationship to experimental data on synaptic plasticity

In biological STDP experiments that induce pairs of pre- and post-synaptic spikes at different time delays, it has been observed that the shape of the plasticity curve changes as a function of the repetition frequency for those spike pairs [207]. The observed effect is that at very low frequencies no change or only LTD occurs, a "classical" STDP window with timing-dependent LTD and LTP is observed at intermediate frequencies around 20 Hz, and at high frequencies of 40 Hz or above only LTP is observed, independently of which spikes comes first.

Although our theoretical model does not explicitly include a stimulation-frequency dependent term like other STDP models (e.g. [79]), we can study empirically the effect of a modification of the frequency of spike-pairing. We simulate this for a single synapse, at which we force pre- and post-synaptic spikes with varying time differences $\Delta t = t_{post} - t_{pre}$, and at fixed stimulation frequencies f of either 1 Hz, 20 Hz, or 40 Hz. Modeling EPSPs as α-kernels with time constants of 1 ms for the rise and 15 ms for the decay, we obtain the low-pass filtered signals $\tilde{y}_i$ as in Eq. (4.17), which grow as EPSPs start to overlap at higher stimulation frequencies. At the time of a post-synaptic spike we compute the synaptic update according to the rule in Eq. (4.18), but keep both the weight and the learning rate fixed (at $w_{ki} = 3.5, c = e^{-5}, \eta = 0.5$) to distinguish timing-dependent from weight-dependent effects.

In Fig. 4.4A we observe that, as expected, at low stimulation frequencies (1 Hz) the standard shape of the complex STDP rule in Eq. (4.18) from Fig. 4.2 is recovered, since there is no influence from previous spikes. The shift towards pure LTD that is observed

in biology [207] would require an additional term that depends on postsynaptic firing rates like in [79], and is a topic of future research. However, note that in biology this shift to LTD was observed only in paired recordings, neglecting the cooperative effect of other synapses, and other studies have also reported LTP at low stimulation frequencies [18]. At higher stimulation frequencies (20 Hz in Fig. 4.4B) the EPSPs from different pre-synaptic spikes start to overlap, which results in larger $\tilde{y}_i$ compared with isolated pre-synaptic spikes. We also see that the LTD part of the STDP window becomes timing-dependent (due to overlapping EPSPs), and thus the shape of the STDP curve becomes similar to standard models of STDP and observed biological data [18, 210]. For even higher stimulation frequencies the STDP window shifts more and more towards LTP (see Fig. 4.4B and C). This is in good accordance with observations in biology [207]. Also in agreement with biological data, the minimum of the update occurs around $\Delta t = 0$, because there the new α-kernel EPSP is not yet effective, and the activation due to previous spikes has decayed maximally.

Another effect that is observed in hippocampal synapses when two neurons are stimulated with bursts, is that the magnitude of LTP is determined mostly by the amount of overlap between the pre- and post-synaptic bursts, rather than the exact timing of spikes [124]. In Fig. 4.4D we simulated this protocol with our continuous-time SEM rule for different onset time-differences of the bursts, and accumulated the synaptic weight updates in response to 50 Hz bursts of 5 pre-synaptic and 4 post-synaptic spikes. We performed this experiment for the same onset time differences used in Fig.3 of [124], and found qualitatively similar results. For long time-differences, when EPSPs have mostly decayed, we observed an LTD effect, which was not observed in biology, but can be attributed to differences in synaptic time constants between biology and simulation.

These results suggest that our STDP rule derived from theoretical principles exhibits several of the key properties of synaptic plasticity observed in nature, depending on the encoding of inputs. This is quite remarkable, since these properties are not explicitly part of our learning rule, but rather emerge from a simpler rule with strong theoretical guarantees. Other phenomenological [34, 154] or mechanistic models of STDP [84] also show some of these characteristics, but come without such theoretical properties. The functional consequence of reproducing such key biological characteristics of STDP is that our new learning rule also exhibits most of the key functional properties of

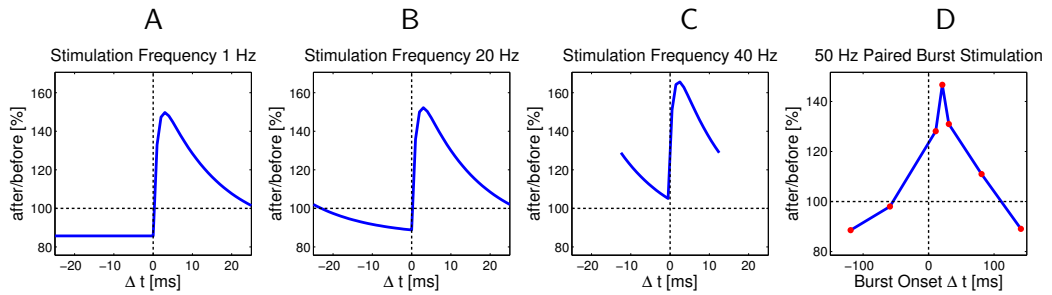

Figure 4.4: Relationship between the continuous-time SEM model and experimental data on synaptic plasticity. A-C: The effect of the continuous-time plasticity rule in Eq. (4.18) at a single synapse for different stimulation frequencies and different time-differences between pre- and post-synaptic spike pairs. Only time-intervals without overlapping pairs are shown. **A**: For very low stimulation frequencies (1 Hz) the standard shape of the complex learning rule from Fig. 4.2 is recovered. **B**: At a stimulation frequency of 20 Hz the plasticity curve shifts more towards LTP, and depression is no longer time independent, due to overlapping EPSPs. **C**: At high stimulation frequencies of 40 Hz or above, the STDP curve shifts towards only LTP, and thus becomes similar to a rate-based Hebbian learning rule. **D**: Cumulative effect of pre- and post-synaptic burst stimulation (50 Hz bursts of 5 pre-synaptic and 4 post-synaptic spikes) with different onset delays of -120, -60, 10, 20, 30, 80 and 140 ms (time difference between the onsets of the post- and pre-synaptic bursts). As in [124], the amount of overlap between bursts determines the magnitude of LTP, rather than the exact temporal order of spikes.

STDP, like e.g. strengthening synapses of inputs that are causally involved in firing the postsynaptic neuron, while pruning the connections that do not causally contribute to postsynaptic firing [1, 209]. At low and intermediate firing rates our rule also shifts the onset of postsynaptic firing towards the start of repeated spike patterns [77, 147, 149], while depressing synapses that only become active for a pattern following the one for which the post-synaptic neuron is responsive. If patterns change quickly, then the stronger depression for presynaptic spikes with small Δt in Fig. 4.4B enhances the capability of the WTA to discriminate such patterns. With simultaneous high frequency stimulation (Fig. 4.4C and D) we observe that only LTP occurs, which is due to the decay of EPSPs not being fast enough to allow depression. In this scenario, the learning rule is less sensitive to timing, and rather becomes a classical Hebbian measure of correlations between pre- and post-synaptic firing rates. However, since inputs are encoded in a population code we can assume that the same neuron is not

continuously active throughout, and so even at high firing rates for active input neurons, the synapses that are inactive during postsynaptic firing will still be depressed, which means that convergence to an equilibrium value is still possible for all synapses.

It is a topic of future research which effects observed in biology can be reproduced with more complex variations of the spike-based EM rule that are also dependent on postsynaptic firing rates, or whether existing phenomenological models of STDP can be interpreted in the probabilistic EM framework. In fact, initial experiments have shown that several variations of the spike-based EM rule can lead to qualitatively similar empirical results for the learned models in tasks where the input spike trains are Poisson at average or high rates over an extended time window (such as in Fig. 4.3). These variations include weight-dependent STDP rules that are inversed in time, symmetrical in time, or have both spike timing-dependent LTD and LTP. Such rules can converge towards the same equilibrium values as the typical causal STDP rule. However, they will behave differently if inputs are encoded through spatio-temporal spike patterns (as in Example 4: Detection of Spatio-Temporal Spike Patterns). Further variations can include short-term plasticity effects for pre-synaptic spikes, as observed and modeled in [68], which induce a stimulation-frequency dependent reduction of the learning rate, and could thus serve as a stabilization mechanism.

4.2.7 Spike-timing dependent LTD

Current models of STDP typically assume a "double-exponential" decaying shape of the STDP curve, which was first used in [210] to fit experimental data. This is functionally different from the shape of the complex STDP curve in Fig. 4.2 and Eq. (4.5), where the LTD part is realized by a constant timing-independent offset.

Although not explicitly covered by the previously presented theory of SEM, the same analytical tools can be used to explain functional consequences of timing-dependent LTD in our framework. Analogous to our approach for the standard SEM learning rule, we develop (in Methods) an extension of the simple step-function STDP rule from Fig. 4.2 with timing-dependent LTD, which is easier to analyze. We then generalize these results towards arbitrarily shaped STDP curves. The crucial result is that as long as the spike-timing dependent LTD rule retains the characteristic inversely-exponential weight-dependent relationship between the strengths of LTP and LTD that was introduced for standard SEM in Eq. (4.5), an equilibrium property similar to Eq. (4.6) still

holds (see Methods for details). Precisely speaking, the new equilibrium will be at the difference between the logarithms of the average presynaptic spiking probabilities *before* and *after* the postsynaptic spike. This shows that spike-timing dependent LTD also yields synaptic weights that can be interpreted in terms of log-probabilities, which can thus be used for inference.

The new rule emphasizes contrasts between the current input pattern and the immediately following activity. Still, the results of the new learning rule and the original rule from Eq. (4.5) in our experiments are qualitatively similar. This can be explained from a stochastic learning perspective: at any point in time the relative spiking probabilities of excitatory neurons in the WTA circuit in Eq. (4.4) depend causally on the weighted sums of preceding presynaptic activities $\tilde{y}_i(t)$. However, they clearly do not depend on future presynaptic activity. Thus, the postsynaptic neuron will learn through SEM to fire for increasingly similar stochastic realizations of presynaptic input $\tilde{y}_i(t)$, whereas the presynaptic activity pattern following a postsynaptic spike will become more variable. In the extreme case where patterns are short and separated by noise, there will be no big difference between input patterns following firing of any of the WTA neurons, and so their relevance for the competition will become negligible.

Experimental evidence shows that the time constants of the LTP learning window are usually smaller than the time constants of the LTD window ([68, 207]), which will further enhance the specificity of the LTP learning as opposed to the LTD part that computes the average over a longer window.

Note that the exponential weight dependence of the learning rule implies a certain robustness towards linearly scaling LTP or LTD strengths, which only leads to a constant offset of the weights. Assuming that the offset is the same for all synapses, this does not affect firing probabilities of neurons in a WTA circuit (see Methods "Weight offsets and positive weights").

4.2.8 Example 2: Learning of probabilistic models for orientation selectivity

We demonstrated in this computer experiment the emergence of orientation selective cells z_k through STDP in the WTA circuit of Fig. 4.1A when the spike inputs encode isolated bars in arbitrary orientations. Input images were generated by the following process: Orientations were sampled from a uniform distribution, and lines of 7 pixels

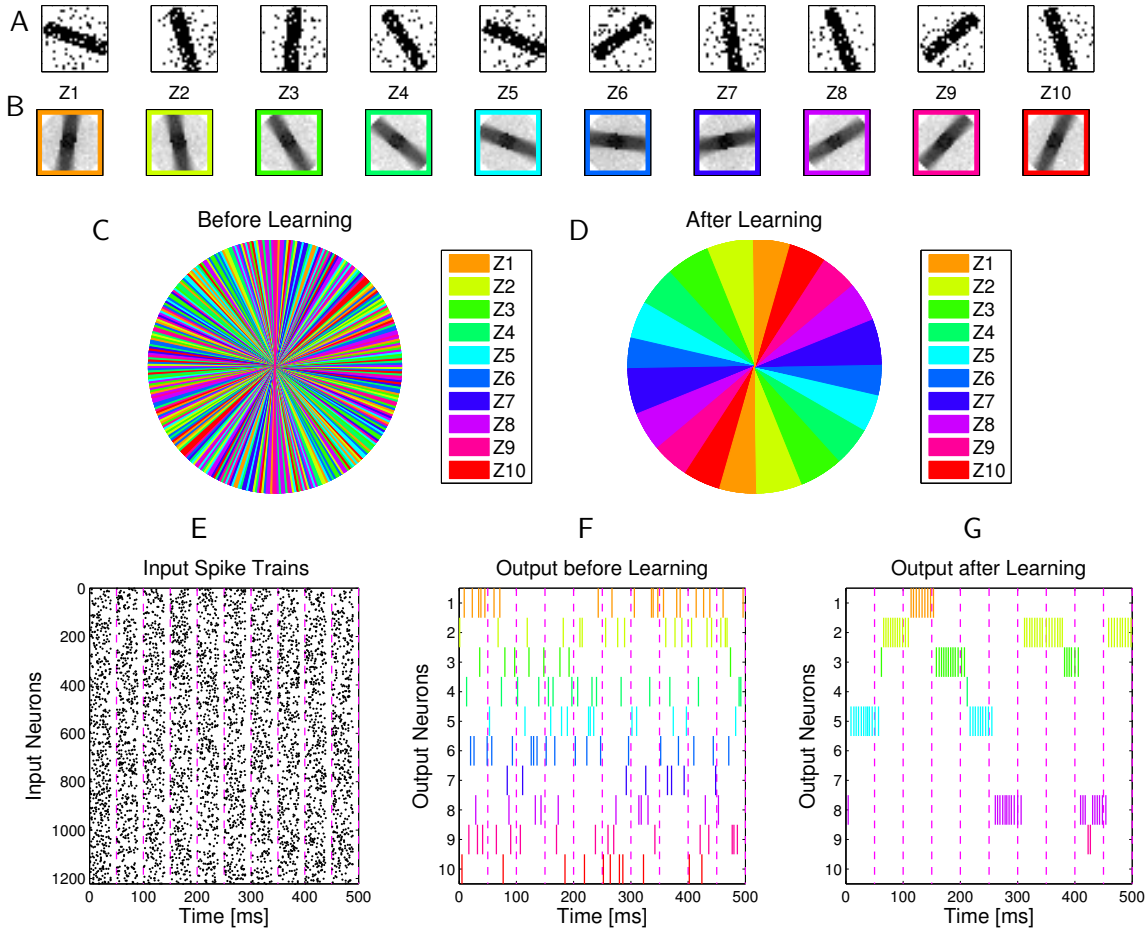

Figure 4.5: Emergence of orientation selective cells for visual input consisting of oriented bars with random orientations. A Examples of 28×28-pixel input images with oriented bars and additional background noise. **B** Internal models (weight vectors of output neurons z_k) that are learned through STDP after the presentation of 4000 input images (each encoded by spike trains for 50 ms, as in Fig.4.3). **C, D** Plot of the most active neuron for 360 images of bars with orientations from 0 to 360° in 1° steps. Colors correspond to the colors of z_k neurons in B. Before training (**C**), the $K = 10$ output neurons fire without any apparent pattern. After training (**D**) they specialize on different orientations and cover the range of possible angles approximately uniformly. **E**: Spike train encoding of the 10 samples in A. **F,G**: Spike trains produced by the $K = 10$ output neurons in response to these samples before and after learning with STDP for 200 s. Colors of the spikes indicate the identity of the output neuron, according to the color code in B.

width were drawn in a 28 x 28 pixel array. We added noise to the stimuli by flipping every pixel with a 10% chance, see Fig. 4.5A. Finally, a circular mask was applied to the images to avoid artifacts from image corners. Spikes trains $\mathbf{y}$ were encoded according to the same population coding principle described in the previous example Fig. 4.3, in this case using a Poisson firing rate of 20 Hz for active units.

After training with STDP for 200 s, presenting 4000 different images, the projection of the learned weight vectors back into the 2D input space (Fig. 4.5B) shows the emergence of 10 models with different orientations, which cover the possible range of orientations almost uniformly. When we plot the strongest responding neuron as a function of orientation (Fig. 4.5C, D), measured by the activity in response to 360 noise-free images of oriented bars in 1° steps, we can see no structure in the response before learning (Fig. 4.5C). However, after unsupervised learning, panel D clearly shows the emergence of continuous, uniformly spaced regions in which one of the z_k neurons fires predominantly. This can also be seen in the firing behavior in response to the input spike trains in Fig. 4.5E, which result from the example images in panel A. Fig. 4.5F shows that the output neurons initially fire randomly in response to the input, and many different z_k neurons are active for one image. In contrast, the responses after learning in panel G are much sparser, and only occasionally multiple neurons are active for one input image, which is the case when the angle of the input image is in between the preferred angles of two output neurons, and therefore multiple models have a non-zero probability of firing.

In our experiment the visual input consisted of noisy images of isolated bars, which illustrates learning of a probabilistic model in which a continuous hidden cause (the orientation angle) is represented by a population of neurons, and also provides a simple model for the development of orientation selectivity. It has previously been demonstrated that similar Gabor-like receptive field structures can be learned with a sparse-coding approach using patches of natural images as inputs [167]. The scenario considered here is thus substantially simplified, since we do not present natural but isolated stimuli. However, it is worth noting that experimental studies have shown that (in mice and ferret) orientation selectivity, but not e.g. direction selectivity, exists in V1 neurons even before eye opening [57, 139]. This initial orientation selectivity develops from innate mechanisms and from internally generated inputs during this phase [57], e.g. retinal waves, which have different, and very likely simpler statistics than natural

stimuli. Our model shows that a WTA circuit could learn orientation selectivity from such simple bar-like inputs, but does not provide an alternative explanation to the results of studies like [167] using natural image stimuli. Although beyond the scope of this paper, we expect that later shaping of selectivity through exposure to natural visual experience would not alter the receptive fields by much, since the neurons have been primed to spike (and thereby trigger plasticity) only in response to a restricted class of local features.

4.2.9 Example 3: Emergent discrimination of handwritten digits through STDP

Spike-based EM is a quite powerful learning principle, as we demonstrate in Fig. 4.6 through an application to a computational task that is substantially more difficult than previously considered tasks for networks of spiking neurons: We show that a simple network of spiking neurons can learn without any supervision to discriminate handwritten digits from the MNIST benchmark dataset [135] consisting of 70,000 samples (30 are shown in Fig. 4.6A). This is one of the most frequently used benchmark tasks in machine learning. It has mostly been used to evaluate supervised or semi-supervised machine learning algorithms [33, 100], or to evaluate unsupervised feature learning approaches [101, 180]. Although the MNIST dataset contains labels (the intended digit) for each sample of a handwritten digit, we deleted these labels when presenting the dataset to the neural circuit of Fig. 4.1A, thereby forcing the $K = 100$ neurons on the output layer to self-organize in a completely unsupervised fashion. Each sample of a handwritten digit was encoded by 708 spike trains over 40 ms (and 10 ms periods without firing between digits to avoid overlap of EPSPs between images), similarly as for the task of Fig. 4.3. Each pixel was represented by two input neurons y_i, one of which produced a Poisson spike train at 40 Hz during these 40 ms. This yielded usually at most one or two spikes during this time window, demonstrating that the network learns and computes with information that is encoded through spikes, rather than firing rates. After 500 s of unsupervised learning by STDP almost all of the output neurons fired more sparsely, and primarily for handwritten samples of just one of the digits (see Fig. 4.6E).

The application to the MNIST dataset had been chosen to illustrate the power of SEM in complex tasks. MNIST is one of the most popular benchmarks in machine

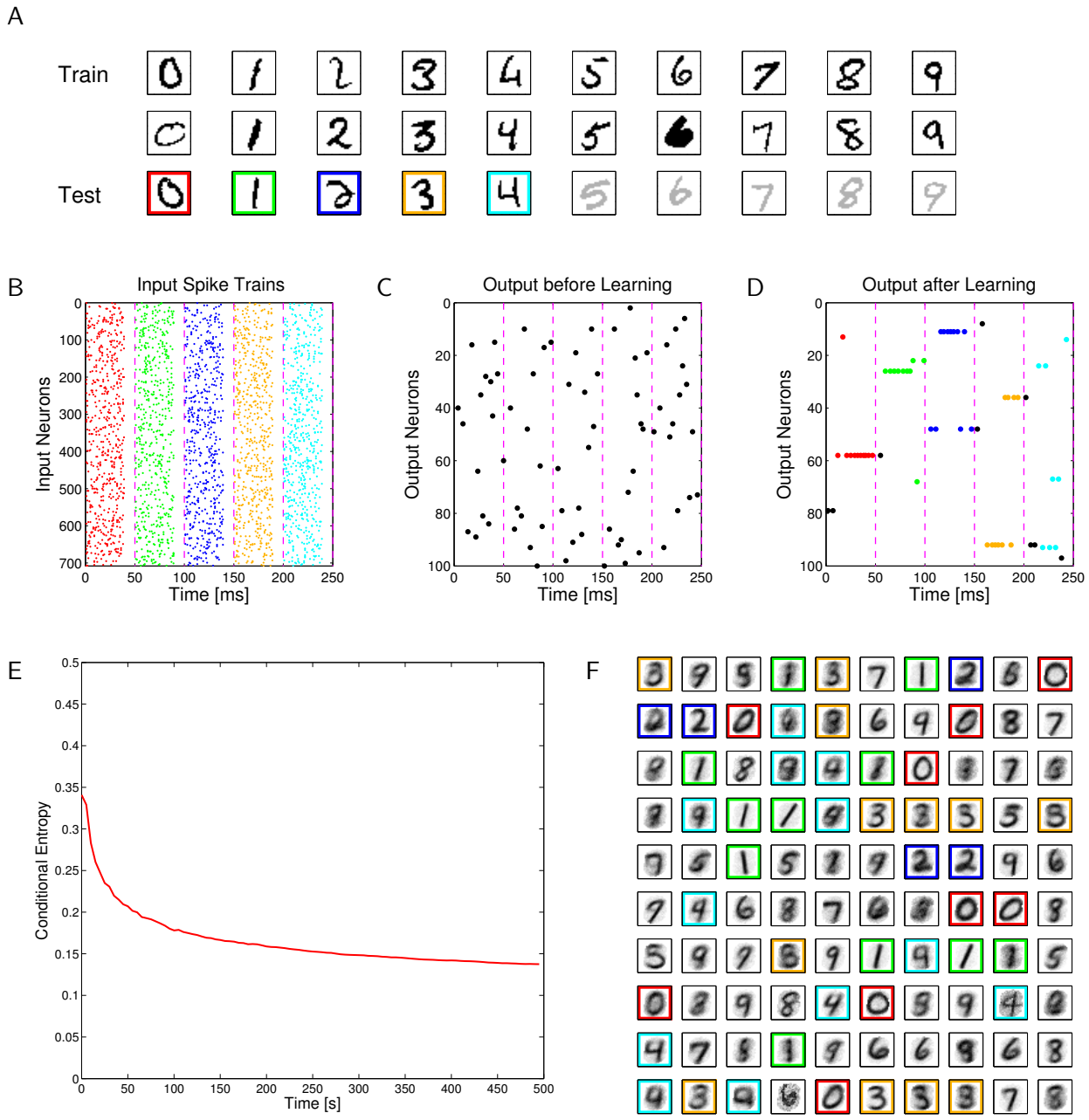

Figure 4.6: Emergent discrimination of handwritten digits through STDP. A: Examples of digits from the MNIST dataset. The third and fourth row contain test examples that had not been shown during learning via STDP. **B**: Spike train encoding of the first 5 samples in the third row of A. Colors illustrate the different classes of digits. **C, D**: Spike trains produced by the $K = 100$ output neurons before *Continued on next page ...*

Continued: Caption for Figure 4.6. and after learning with STDP for 500 s. Colored spikes indicate that the class of the input and the class for which the neuron is mostly selective (based on human classification of its generative model shown in F) agree, otherwise spikes are black. **E**: Temporal evolution of the self-organization process of the 100 output neurons (for the complex version of STDP-curve shown in Fig. 4.1B), measured by the conditional entropy of digit labels under the learned models at different time points. **F**: Internal models generated by STDP for the 100 output neurons after 500 s. The network had not received any information about the number of different digits that exist and the colors for different ways of writing the first 5 digits were assigned by the human supervisor. On the basis of this assignment the test samples in row 3 of panel A had been recognized correctly.

learning, and state-of-the-art methods achieve classification error rates well below 1%. The model learned by SEM can in principle also be used for classification, by assigning each neuron to the class for which it fires most strongly. However, since this is an unsupervised method, not optimized for classification but for learning a generative model, the performance is necessarily worse. We achieve an error rate of 19.86% on the 10-digit task on a previously unseen test set. This compares favorably to the 21% error that we obtained with a standard machine learning approach that directly learned the mixture-of-multinomials graphical model in Fig. 4.1B with a batch EM algorithm. This control experiment was not constrained by a neural network architecture or biologically plausible learning, but instead mathematically optimized the parameters of the model in up to 200 iterations over the whole training set. The batch method achieves a final conditional entropy of 0.1068, which is slightly better than the 0.1375 final result of the SEM approach, and shows that better performance on the classification task does not necessarily mean better unsupervised model learning.

4.2.10 Example 4: Detection of Spatio-Temporal Spike Patterns

Our final application demonstrates that the modules for Bayesian computation that emerge in WTA circuits through STDP can not only explain the emergence of feature maps in primary sensory cortices like in Fig. 4.5, but could also be viewed as generic computational units in generic microcircuits throughout the cortex. Such generic microcircuit receives spike inputs from many sources, and it would provide a very useful computational operation on these if it could autonomously detect repeatedly occurring spatio-temporal patterns within this high-dimensional input stream, and report their occurrence through a self-organizing sparse coding scheme to other microcircuits. We

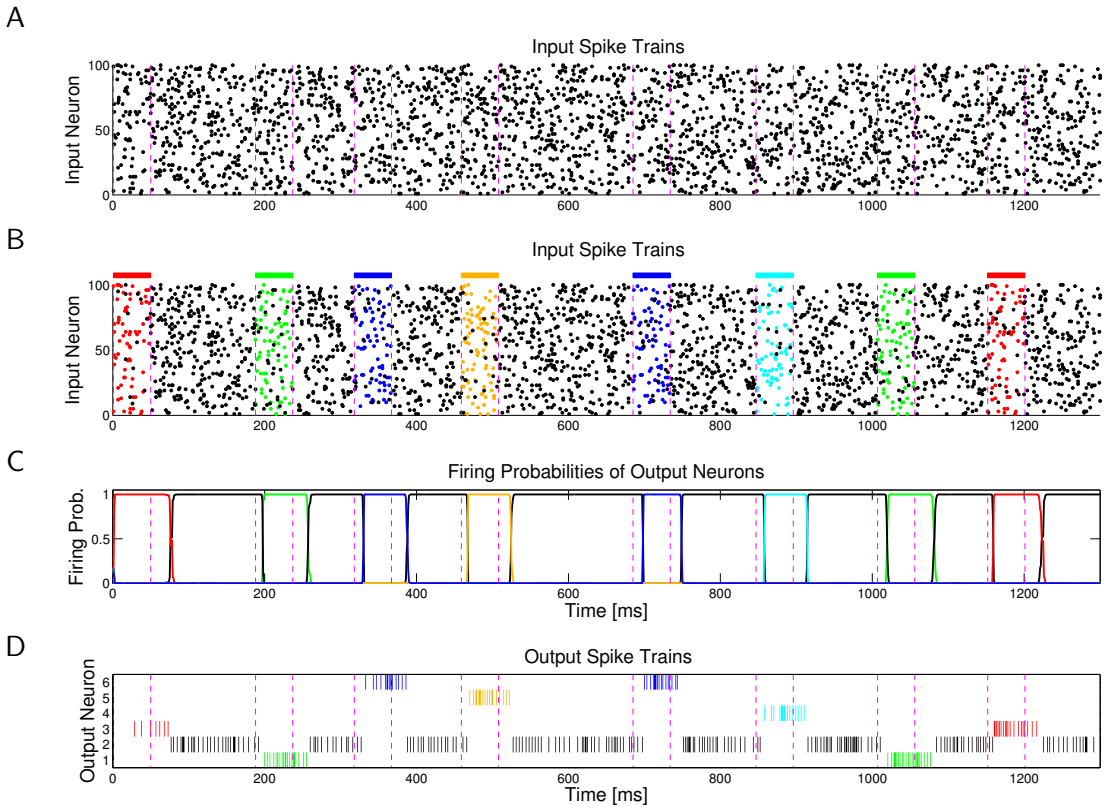

Figure 4.7: Output neurons self-organize via STDP to detect and represent spatio-temporal spike patterns. A: Sample of the Poisson input spike trains at 20 Hz (only 100 of the 500 input channels are shown). Dashed vertical lines mark time segments of 50 ms length where spatio-temporal spike patterns are embedded into noise. **B**: Same spike input as in A, but spikes belonging to five repeating spatio-temporal patterns (frozen Poisson spike patterns at 15 Hz) are marked in five different colors. These spike patterns are superimposed by noise (Poisson spike trains at 5 Hz), and interrupted by segments of pure noise of the same statistics (Poisson spike trains at 20 Hz) for intervals of randomly varying time lengths. *Continued on next page...*

Continued caption for Fig. 4.7: **C, D**: Firing probabilities and spike outputs of 6 output neurons (z-neurons in Fig. 4.1A) for the spike input shown in A, after applying STDP for 200 s to continuous spike trains of the same structure (without any supervision or reward). These 6 output neurons have self-organized so that 5 of them specialize on one of the 5 spatio-temporal patterns. One of the 6 output neurons (firing probability and spikes marked in black) only responds to the noise between these patterns. The spike trains in A represent test inputs, that had never been shown during learning.

have created such input streams with occasionally repeated embedded spike patterns for the computer experiment reported in Fig. 4.7. Fig. 4.7D demonstrates that sparse output codes for the 5 embedded spike patterns emerge after applying STDP in a WTA circuit for 200 s to such input stream. Furthermore, we show in the Supplement that these sparse output codes generalize (even without any further training) to time-warped versions of these spike patterns.

Even though our underlying probabilistic generative model (Fig. 4.1B) does not include time-dependent terms, the circuit in this example performs inference over time. The reason for this is that synapses that were active when a neuron fired become reinforced by STDP, and therefore make the neuron more likely to fire again when a similar spatial pattern is observed. Since we use EPSPs that smoothly decay over time, one neuron still sees a trace of previous input spikes as it fires again, and thus different spatial patterns within one reoccurring spatio-temporal pattern are recognized by the same neuron. The maximum length for such patterns is determined by the time constants of EPSPs. With our parameters (1 ms rise, 15 ms decay time constant) we were able to recognize spike patterns up to 50-100 ms. For longer spatio-temporal patterns, different neurons become responsive to different parts of the pattern. The neuron that responds mostly to noise in Figs. 4.7D did not learn a specific spatial pattern, and therefore wins by default when none of the specialized neurons responds. Similar effects have previously been described [147, 148], but for different neuron models, classical STDP curves, and not in the context of probabilistic inference.

For this kind of task, where also the exact timing of spikes in the patterns matters (which is not necessarily the case in the examples in Figs. 4.3, 4.5, and 4.6, where input neurons generate Poisson spike trains with different rates), we found that the shape of the STDP kernel plays a larger role. For example, a time-inverted version of the SEM rule, where pre-before-post firing causes LTD instead of LTP, cannot learn this kind of task, because once a neuron has learned to fire for a sub-pattern of the input, its firing onset is shifted back in time, rather than forward in time, which happens with standard SEM, but also with classical STDP [77, 147]. Instead, with a time-inverted SEM rule, different neurons would learn to fire stronger for the offsets of different patterns.

Such emergent compression of high-dimensional spike inputs into sparse low-dimensional spike outputs could be used to merge information from multiple sensory modalities, as well as from internal sources (memory, predictions, expectations, etc.), and to report

the co-occurrence of salient events to multiple other brain areas. This operation would be useful from the computational perspective no matter in which cortical area it is carried out. Furthermore, the computational modules that we have analyzed can easily be connected to form networks of such modules, since their outputs are encoded in the same way as their inputs: through probabilistic spiking populations that encode for abstract multinomial variables. Hence the principles for the emergence of Bayesian computation in local microcircuits that we have exhibited could potentially also explain the self-organization of distributed computations in large networks of such microcircuits.

4.3 Discussion

We have shown that STDP induces a powerful unsupervised learning principle in networks of spiking neurons with lateral inhibition: spike-based Expectation Maximization. Each application of STDP can be seen as a move in the direction of the M-step in a stochastic online EM algorithm that strives to maximize the log-likelihood $\log p(\mathbf{y}|\mathbf{w})$ of the spike input $\mathbf{y}$. This is equivalent to the minimization of the Kullback-Leibler divergence between the true distribution $p^*(\mathbf{y})$ of spike inputs, and the generative model $p(\mathbf{y}|\mathbf{w})$ that is implicitly represented by the WTA circuit from the Bayesian perspective. This theoretically founded principle guarantees that iterative applications of STDP to different spike inputs do not induce a meaningless meandering of the synaptic weights $\mathbf{w}$ through weight space, but rather convergence to at least a local optimum in the fitting of the model to the distribution $p^*(\mathbf{y})$ of high-dimensional spike inputs $\mathbf{y}$. This generation of an internal model through STDP provides the primary component for the self-organization of Bayesian computation. We have shown that the other component, the prior, results from a simple rule for use-dependent adaptation of neuronal excitability. As a consequence, the firing of a neuron z_k in a stochastic WTA circuit (Fig. 4.1A) can be viewed as sampling from the posterior distribution of hidden causes for high-dimensional spike inputs $\mathbf{y}$ (and simultaneously as the E-step in the context of online EM): A prior (encoded by the thresholds w_{k0} of the neurons z_k) is multiplied with a likelihood (encoded through an implicit generative distribution defined by the weights $w_{k1}, \ldots, w_{kn}$ of these neurons z_k), to yield through the firing probabilities of the neurons z_k a representation of the posterior distribution of hidden causes for the

current spike input $\mathbf{y}$. The multiplications and the divisive normalization that are necessary for this model are carried out by the linear neurons in the log-scale. This result is then transformed into an instantaneous firing rate, assuming an exponential relationship between rate and the membrane potential [113]. It is important that the neurons z_k fire stochastically, i.e., that there exists substantial trial-to trial variability, since otherwise they could not represent a probability distribution. Altogether our models supports the view that probability distributions, rather than deterministic neural codes, are the primary units of information in the brain, and that computational operations are carried out on probabilities, rather than on deterministic bits of information.

Following the "probabilistic turn" in cognitive science [86, 87, 164] and related hypotheses in computational neuroscience [54, 127, 183], probabilistic inference has become very successful in explaining behavioral data on human reasoning and other brain functions. Yet, it has remained an important open problem how networks of spiking neurons can learn to implement those probabilistic inference operations and probabilistic data structures. The soft WTA model presented in this article provides an answer for the case of Bayesian inference and learning in a simple graphical model, where a single hidden cause has to be inferred from bottom-up input. Although this is not yet a mechanism for learning to perform general Bayesian inference in arbitrary graphical models, it clearly is a first step into that direction. Importantly, the encoding of posterior distributions through spiking activity of the neurons z_k in a WTA circuit is perfectly compatible with the assumed input encoding from external variables x_j into spiking activity in $\mathbf{y}$. Thus, the interpretation of spikes from output neurons z_k as samples of the posterior distributions over hidden variables in principle allows for using these spikes as input for performing further probabilistic inference.

This compatibility of input and output codes means that SEM modules could potentially be hierarchically and/or recurrently coupled in order to serve as inputs of one another, although it remains to be shown how this coupling affects the dynamics of learning and inference. Future research will therefore address the important questions whether interconnected networks of modules for Bayesian computation that emerge through STDP can provide the primitive building blocks for probabilistic models of cortical computation. Previous studies [172, 192] have shown that interconnected networks of WTA modules are indeed computationally very powerful. In particular, [27, 172] have recently shown how recurrently connected neurons can be designed to

perform neural sampling, an approach in which time-independent probability distributions can be represented through spiking activity in recurrent neural networks. The question how salient random variables come to be represented by the firing activity of neurons has remained open. This paper shows that such representations may emerge autonomously through STDP.

A prediction for networks of hierarchically coupled SEM modules would be that more and more abstract hidden causes can be learned in higher layers such as it has been demonstrated in machine learning approaches using Deep Belief Networks [101] and more recently in Deep Boltzmann Machines (DBM) [196]. This effect would correspond to the emergence of abstract feature selectivity in higher visual areas of primates (e.g. face-selective cells in IT, [49]). The hierarchical structure, however, that would result from such deeply organized SEM-modules is more reminiscent of a Deep Sum-Product Network [177], a recently presented new architecture, which has a much simpler learning dynamics but arguably a similar expressive power as DBM. In addition, with a consistent input encoding, associations between different sensory modalities could be formed by connecting inputs from different low-level or high-level sources to a single SEM.

Importantly, while the discussion above focused only on the representation of complex stimuli by neurons encoding abstract hidden causes, SEM can also be an important mechanism for fast and reliable reinforcement learning or decision making under uncertainty. Preprocessing via single or multiple SEM circuits provides an abstraction of the state of the organism, which is much lower-dimensional than the complete stream of individual sensory signals. Learning a behavioral strategy by reading out such behaviorally relevant high-level state signals and mapping them into actions could therefore speed up learning by reducing the state space. In previous studies [158, 173] we have shown how optimal strategies can be learned very fast by simple local learning rules for reinforcement learning or categorization, if a preprocessing of input signals based on probabilistic dependencies is performed. SEM would be a suitable unsupervised mechanism for learning such preprocessing networks for decision making.

We also have shown that SEM is a very powerful principle that endows networks of spiking neurons to solve complex tasks of practical relevance (see e.g. Fig. 4.6), and as we have shown, their unsupervised learning performance is within the range of conventional machine learning approaches. Furthermore, this could be demonstrated for

computations on spike inputs with an input dimension of about 1000 presynaptic neurons $y_1, \ldots, y_n$, a number that approaches the typical dimension of the spike input that a cortical neuron receives. A very satisfactory aspect is that this high computational performance can be achieved by networks of spiking neurons that learn completely autonomously by STDP, without any postulated teacher or other guidance. This could benefit the field of neuromorphic engineering [107, 112, 202], which develops dedicated massively parallel and very efficient hardware for emulating spiking neural networks and suitable plasticity rules. The link between spiking neuron models and plasticity rules and established machine learning concepts provides a novel way of installing well-understood Bayesian inference and learning mechanisms on neuromorphic hardware. First steps towards implementing SEM-like rules in different types of neuromorphic hardware have been taken.

4.3.1 Prior related work

A first model for competitive Hebbian learning paradigm in non-spiking networks of neurons had been introduced in [191]. They analyzed a Hebbian learning rule in a hard WTA network and showed that there may exist equilibrium states, in which the average change of all weight values vanishes for a given set of input patterns. They showed that in these cases the weights adopt values that are proportional to the conditional probability of the presynaptic neuron being active given that the postsynaptic unit wins (rather than the log of this conditional probability, as in our framework). [162] showed that the use of a soft competition instead of a hard winner assignment and corresponding average weight updates lead to an exact gradient ascent on the log-likelihood function of a generative model of a mixture of Gaussians. However, these learning rules had not yet been analyzed in the context of EM.

Stochastic approximation algorithms for expectation maximization [44] were first considered in [31], incremental and on-line EM algorithms with soft-max competition in [114, 155, 163]. A proof of the stochastic approximation convergence for on-line EM in exponential family models with hidden variables was shown in [198]. They developed a sophisticated schedule for the learning rate in this much more general model, but did not yet consider individual learning rates for different weights.

[210] initiated the investigation of STDP in the context of unsupervised competitive Hebbian learning and demonstrated that correlations of input spike trains can be

learned in this way. They also showed that this leads to a competition between the synapses for the control of the timing of the postsynaptic action potential. A similar competition can also be observed during learning in our model, since our learning rule automatically drives the weights towards satisfying the normalization conditions in Eq. (4.10).

[200] present a network and learning model that is designed to perform Independent Component Analysis (ICA) with spiking neurons through STDP and intrinsic plasticity. The mixture model of independent components can also be formulated as a generative model, and the goal of ICA is to find the optimal parameters of the mixing matrix. It has been shown that also this problem can be solved by a variant of Expectation Maximization [38], so there is some similarity to the identification of hidden causes in our model.

Recently, computer experiments in [90, 91] have used STDP in the context of WTA circuits to achieve a clustering of input patterns. Their STDP rules implements linear updates, independent of the current weight values, mixed with a homeostasis rule to keep the sum of all weights constant and every weight between 0 and 1. This leads to weights that are roughly proportional to the probability of the presynaptic neuron's firing given that the post-synaptic neuron fires afterwards. The competition between the output neurons is carried out as hard-max. In [91] the 4 output neurons learn to differentiate the 4 presented patterns and smoothly interpolate new rotated input patterns, whereas in [90] 48 neurons learn to differentiate characters in a small pixel raster. [90] uses a STDP rule where both LTP and LTD are modeled as exponentially dependent on the time difference. However, the very specific experimental setting with synchronous regular firing of the input neurons makes it difficult to generalize their result to more general input spike trains. No theoretical analysis is provided in [91] or [90], but their experimental results can be explained by our SEM approach. Instead of adding up logs of conditional probabilities and performing the competition on the exponential of the sums, they sum up the conditional probabilities directly and use this sum of probabilities for the competition. This can be seen as a linear approximation of SEM, especially under the additional normalization conditions that they impose by homeostasis rules.

It has previously been shown that spike patterns embedded in noise can be detected by STDP [77, 147, 149]. Competitive pattern learning through STDP has recently

been studied in [148]. They simulate a deterministic version of a winner-take-all circuit consisting of a fixed number of neurons, all listening to the same spiking input lines and connected to each other with a strong inhibition. The STDP learning rule that they propose is additive and weight-independent. Just like our results, they also observe that different neurons specialize on different fixed repeated input pattern, even though the repeated patterns are embedded in spiking noise such that the mean activity of all inputs remains the same throughout the learning phase. Additionally they show that within each pattern the responsible neuron tries to detect the start of the pattern. In contrast to our approach they do not give any analysis of convergence guarantees, nor does their model try to build a generative probabilistic model of the input distribution.

[181, 182, 236] investigated the possibility to carry out Bayesian probabilistic computations in recurrent networks of spiking neurons, both using probabilistic population codes. They showed that the ongoing dynamics of belief propagation in temporal Bayesian models can be represented and inferred by such networks, but they do not exhibit any neuronal plausible learning mechanism. [143] presented another approach to Bayesian inference using probabilistic population codes, also without any learning result.

An interesting complementary approach is presented in [45, 46], where a single neuron is modeled as hidden Markov model with two possible states. This approach has the advantage, that the instantaneous synaptic input does not immediately decide the output state, but only incrementally influences the probability for switching the state. The weights and the temporal behavior can be learned online using local statistics. The downside of this approach is that this hidden Markov model can have only two states. In contrast, the SEM approach can be applied to networks with any number of output neurons.

In [93] it was shown that a suitable rule for supervised spike-based learning (the Tempotron learning rule) can be used to train a network to recognize spatio-temporal spike patterns. This discriminative learning scheme enables the recognizing neuron to focus on the most discriminative segment of the pattern. In contrast, our generative unsupervised learning scheme drives the recognizing neuron to generalize and spike many times during the whole pattern, and thus learns the spatial average activity pattern. The conductance based approach of [93] differs drastically from our method (and the results shown in the Supplement) insofar as here only STDP was used (focusing

on average spatial patterns), no supervision was involved, and the time-warped input pattern had never been shown during training.

An alternative approach to implement the learning of generative probabilistic models in spiking neuronal networks is given in [25, 186]. Both approaches are based on the idea to model a sequence of spikes in a Hidden-Markov-Model-like probabilistic model and learn the model parameters through different variants of EM, in which a sequence of spikes represents one single sample of the model's distribution. Due to the explicit incorporation of inference over time, these models are more powerful than ours and thus require non-trivial, non-local learning mechanisms.

4.3.2 Experimentally testable predictions of the proposed model

Our analysis has shown, that STDP supports the creation of internal models and implements spike-based EM if changes of synaptic weights depend in a particular way on the current value of the weight: Weight potentiation depends in an inversely exponential manner on the current weight (see Eq. (4.5)). This rule for weight potentiation (see Fig. 4.8A) is consistent with all published data on this dependence: Fig. 5 in [18] and Fig. 5C in [207] for STDP, as well as Fig. 10 in [140] and Fig. 1 in [153] for other protocols for LTP induction. One needs to say, however, that these data exhibit a large trial-to-trial variability, so that it is hard to infer precise quantitative laws from them. On the other hand, the applications of STDP that we have examined in Fig. 4.3 - 4.7 work almost equally well if the actual weight increase varies by up to 100% from the weight increase proposed by our STDP rule (see open circles in Fig. 4.8A). The resulting distribution of weight increases matches qualitatively the above mentioned experimental data quite well.

The prediction of our model for the dependence of the amount of weight depression on the current weight is drastically different: Even though we make the strong simplification that the depression part of the STDP rule is independent of the time difference between pre- and postsynaptic spike, the formulation in Eq. (4.5) makes the assumption, that the amount of the depression should be independent of the current weight value. It is this contrast between an exponential dependency for LTP and a constant LTD which makes the weight converge to the logarithm of the conditional presynaptic firing probability in Eq. (4.6). In experiments this dependency has been

investigated in-vitro [207]. There it has been found that the *percentage* of weight depression under STDP is independent of the current weight, which implies that the amount of depression is linear in the current weight value. This seems to contradict the presented learning rule. However, the key property that is needed for the desired equilibrium condition is the ratio between LTP and LTD. So the equilibrium proof in Eq. (4.28) remains unchanged if Δw_{ki} is multiplied (for potentiation and depression) by some arbitrary function $f(w_{ki})$ of the current weight value. Choosing for example $f(w_{ki}) = w_{ki}$ yields a depression whose percentage is independent of the initial value, which would be consistent with the above mentioned in-vitro data [207]. The resulting dependence for potentiation is plotted in Fig. 4.8B. Since this curve is very similar to that of Fig. 4.8A, the above mentioned experimental data for potentiation are too noisy to provide a clear vote for one of these two curves. Thus more experimental data are needed for determining the dependence of weight potentiation on the initial weight. Whereas the relevance of this dependency had previously not been noted, our analysis suggests that such a contrast it is in fact essential for the capability of STDP to create internal models for high-dimensional spike inputs.

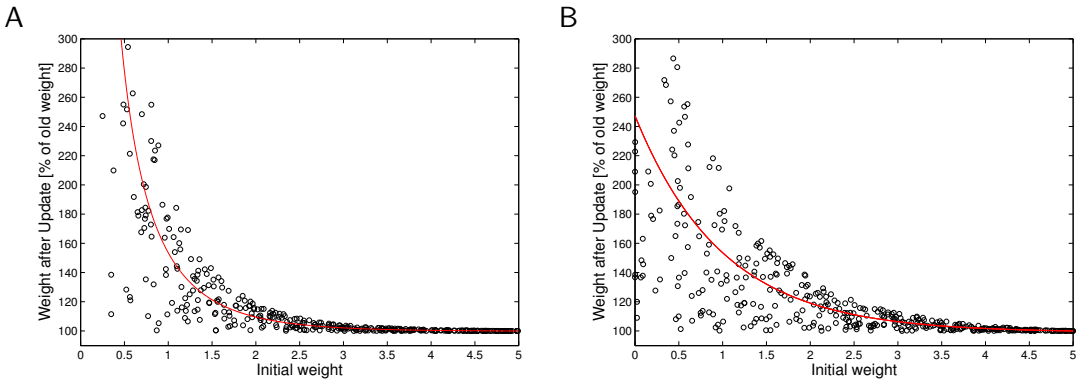

Figure 4.8: Ideal dependence of weight potentiation under STDP on the initial value of the weight (solid lines). Open circles represent results of samples from this ideal curve with 100% noise, that can be used in the previously discussed computer experiments with almost no loss in performance. **A**: Dependence of weight potentiation on initial weight according to the STDP rule in Eq. (4.5). **B**: Same with an additional factor w.

Our analysis has shown, that if the excitability of neurons is also adaptive, with a rule as in Eq. (4.7) that is somewhat analogous to that for synaptic plasticity, then

neurons can also learn appropriate priors for Bayesian computation. Several experimental studies have already confirmed, that the intrinsic excitability of neurons does in fact increase when they are more frequently activated [35], see [43], [37] and [29] for reviews. But a quantitative study, which relates the resulting change in intrinsic excitability to its initial value, is missing.

Our model proposes that pyramidal neurons in cortical microcircuits are organized into stochastic WTA circuits, that together represent a probability distribution. This organization is achieved by a suitably regulated common inhibitory signal, where the inhibition follows the excitation very closely. Such instantaneous balance between excitation and inhibition was described by [166]. A resulting prediction of the WTA structure is that the firing activity of these neurons is highly de-correlated due to the inhibitory competition. In contrast to previous experimental results, that reported higher correlations, it has recently been confirmed in [56] for the visual cortex of awake monkey that nearby neurons, even though they share common input show extremely low correlations.

Another prediction is that neural firing activity especially for awake animals subject to natural stimuli is quite sparse, since only those neurons fire whose internal model matches their spike input. A number of experimental studies confirm this predictions (see [168] for a review). Our model also predicts, that the neural firing response to stimuli exhibits a fairly high trial-to-trial variability, as is typical for drawing repeated samples from a posterior distribution (unless the posterior probability is close to 0 or 1). A fairly high trial-to-trial variability is a common feature of most recordings of neuronal responses (see e.g. [120], Fig. 1B in [161]; a review is provided in [58]). In addition, our model predicts that this trial-to-trial variability decreases for repeatedly occurring natural stimuli (especially if this occurs during attention) and discrimination capability improves for these stimuli, since the internal models of neurons are becoming better fitted to their spike input during these repetitions ("sharpening of tuning"), yielding posterior probabilities closer to 1 or 0 for these stimuli. These predictions are consistent with a number of experimental data related to perceptual learning [75, 76], and with the evolution of neuronal responses to natural scenes that were shown repeatedly in conjunction with nucleus basalis stimulation [80].

In addition our model predicts that if the distribution of sensory inputs changes, the organization of codes for such sensory inputs also changes. More frequently occurring

sensory stimuli will be encoded with a finer resolution (see [42] for a review of related experimental data). Furthermore in the case of sensory deprivation (see [150]) our model predicts that neurons that used to encode stimuli which no longer occur will start to participate in the encoding of other stimuli.

We have shown in Fig. 4.3 that an underlying background oscillation on neurons that provide input to a WTA circuit speeds up the learning process, and produces more precise responses after learning. This result predicts that cortical areas that collaborate on a common computational task, especially under attention, exhibit some coherence in their LFP. This has already been shown for neurons in close proximity [145] but also for neurons in different cortical areas [225, 226].

If one views the modules for Bayesian computation that we have analyzed in this article as building blocks for larger cortical networks, these networks exhibit a fundamental difference to networks of neurons: Whereas a neuron needs a sufficiently strong excitatory drive in order to reach its firing threshold, the output neurons z of a stochastic WTA circuit according to our model in Eq. (4.3) are firing already on their own - even without any excitatory drive from the input neuron y (due to assumed background synaptic inputs; modeled in our simulations by an Ornstein-Uhlenbeck process, as suggested by in-vivo data [50]). Rather, the role of the input from the y-neurons is to modulate which of the neurons in the WTA circuit fire. One consequence of this characteristic feature is that even relatively few presynaptic neurons y can have a strong impact on the firing of the z-neurons, provided the z-neurons have learned (via STDP) that these y-neurons provide salient information about the hidden cause for the total input $\mathbf{y}$ from all presynaptic neurons. This consequence is consistent with the surprisingly weak input from the LGN to area V1 [20, 52, 146]. It is also consistent with the recently found exponential distance rule for the connection strength between cortical areas [146]. This rule implies that the connection strength between distal cortical areas, say between primary visual cortex and PFC, is surprisingly weak. Our model suggests that these weak connections can nevertheless support coherent brain computation and memory traces that are spread out over many, also distal, cortical areas.

Apart from these predictions regarding aspects of brain computation on the microscale and macroscale, a primary prediction of our model is that complex computations in cortical networks of neurons - including very efficient and near optimal processing of uncertain information - are established and maintained through STDP, on

the basis of genetically encoded stereotypical connection patterns (WTA circuits) in cortical microcircuits.

4.4 Methods

According to our input model, every external multinomial variable x_j, with $j = 1, \ldots, m$ is encoded through a group G_j of neurons y_i, with $i \in G_j$. The generative model $p(\mathbf{x}|\theta)$ from Fig. 4.1B is implicitly encoded in the WTA circuit of Fig. 4.1A with K excitatory neurons z_k by:

$$p(\mathbf{y}|\mathbf{w}) = \frac{1}{Z} \cdot \sum_{k=1}^{K} \left[e^{w_{k0}} \cdot \prod_{j=1}^{m} \prod_{i \in G_j} e^{w_{ki} \cdot [x_j = v(i)]} \right] = \frac{1}{Z} \sum_{k=1}^{K} e^{w_{k0} + \sum_{i=1}^{n} w_{ki} \cdot y_i} \tag{4.19}$$

where $[x_j = v(i)]$ is the binary indicator function of x_j taking on value $v(i)$. In the generative model $p(\mathbf{y}|\mathbf{w})$ we define the binary variables y_i and set $y_i = 1$ if y_i represents the value $v(i)$ of the multinomial variable x_j (with j s.t. $i \in G_j$) and $x_j = v(i)$, otherwise $y_i = 0$. The sets G_j represent a partition of $\{1, \ldots, n\}$, thus $\prod_{j=1}^{m} \prod_{i \in G_j} e^{w_{ki} y_i}$ and the form $\prod_{i=1}^{n} e^{w_{ki} y_i}$ used in Eq. (4.9) are equivalent expressions. The value of the normalization constant Z can be calculated explicitly as

$$Z = \sum_{k=1}^{K} e^{w_{k0}} \prod_{j=1}^{m} Z_{kj} \quad , \qquad \text{with} \quad Z_{kj} = \sum_{i \in G_j} e^{w_{ki}} \quad . \tag{4.20}$$

This generative model can be rewritten as a mixture distribution with parameters π_k and μ_{ki}:

$$p(\mathbf{y}|\mathbf{w}) = \sum_{k=1}^{K} p(\mathbf{y}, k|\mathbf{w}) = \sum_{k=1}^{K} \left[\pi_k \cdot \prod_{i=1}^{n} \mu_{ki}^{y_i} \right] \quad , \tag{4.21}$$

$$\pi_k = p(k|\mathbf{w}) = e^{w_{k0}} \frac{\prod_{j=1}^{m} Z_{kj}}{Z} \tag{4.22}$$

$$\mu_{ki} = p(y_i = 1|k, \mathbf{w}) = \frac{e^{w_{ki}}}{Z_{kj}} \quad \text{with } i \in G_j \quad . \tag{4.23}$$

In order to show how the constants Z_{kj} cancel out we write the full joint distribution of $\mathbf{y}$ and the "hidden cause" k as the product of the prior $p(k|\mathbf{w})$ and the likelihood

$p(\mathbf{y}|k,\mathbf{w})$:

$$p(\mathbf{y},k|\mathbf{w}) = p(k|\mathbf{w}) \cdot p(\mathbf{y}|k,\mathbf{w}) \tag{4.24}$$

$$= e^{w_{k0}} \frac{\prod_{j=1}^{m} Z_{kj}}{Z} \cdot \prod_{i=1}^{n} \left(\frac{e^{w_{ki}}}{Z_{kj}} \right)^{y_i} \qquad \text{with } j \text{ such that } i \in G_j \tag{4.25}$$

$$= e^{w_{k0}} \frac{\prod_{j=1}^{m} Z_{kj}}{Z} \cdot \prod_{j=1}^{m} \left(\frac{1}{Z_{kj}} \prod_{i \in G_j} (e^{w_{ki}})^{y_i} \right) \tag{4.26}$$

$$= \frac{1}{Z} e^{w_{k0}} \cdot \prod_{i=1}^{n} e^{w_{ki} y_i} \quad . \tag{4.27}$$

Under the normalization conditions in Eq. (4.10) the parameters of the mixture distribution simplify to $\mu_{ki} = e^{w_{ki}}$ and $\pi_k = e^{w_{k0}}$, since all $Z_{kj} = 1$ and $Z = 1$.

The generative model in Eq. (4.24) is well defined only for vectors $\mathbf{y}$, such that there is exactly one "1" entry per group G_j. However, in the network model with rectangular, renewable EPSPs, there are time intervals where $\mathbf{y}(t)$ may violate this condition, if the interval between two input spikes is longer that σ. It is obvious from Eq. (4.24) that this has the effect of dropping all factors representing x_j, since this results in an exponent of 0. Under proper normalization conditions (or at least if all Z_{kj} have identical values), this drop of an entire input group in the calculation of the posterior in Eq. (4.11) is identical to performing inference with unknown x_j (see 'Impact of missing input values'). Eq. (4.11) holds aslong as there are no two input spikes from different neurons within the same group closer than σ, which we have assumed for the simple input model with rectangular, renewable EPSPs.

Equilibrium condition

We will now show that all equilibria of the stochastic update rule in Eq. (4.5) and Eq. (4.7), i.e., all points where $\mathrm{E}_{p^*_{\mathbf{w}}}[\Delta \mathbf{w}] = \mathbf{0}$, exactly match the implicit solution conditions in Eq. (4.46), and vice versa:

$$\begin{aligned}
\mathrm{E}[\Delta w_{ki}] = \mathbf{0} \;\Leftrightarrow \quad & p^*_{\mathbf{w}}(y_i{=}1|z_k{=}1)(e^{-w_{ki}} - 1) - p^*_{\mathbf{w}}(y_i{=}0|z_k{=}1) = 0 \\
\Leftrightarrow \quad & p^*_{\mathbf{w}}(y_i{=}1|z_k{=}1)(e^{-w_{ki}} - 1) + p^*_{\mathbf{w}}(y_i{=}1|z_k{=}1) - 1 = 0 \\
\Leftrightarrow \quad & p^*_{\mathbf{w}}(y_i{=}1|z_k{=}1)e^{-w_{ki}} = 1 \\
\Leftrightarrow \quad & w_{ki} = \log p^*_{\mathbf{w}}(y_i{=}1|z_k{=}1) \quad .
\end{aligned} \tag{4.28}$$

Analogously, one can show that $\mathrm{E}_{p^*_{\mathbf{w}}}[\Delta w_{k0}] = 0 \Leftrightarrow w_{k0} = \log p^*_{\mathbf{w}}(z_k = 1)$. Note that this result implies that the learning rule in Eq. (4.5) and Eq. (4.7) has no equilibrium points outside the normalization conditions in Eq. (4.10), since all equilibrium points fulfill the implicit solutions condition in Eq. (4.46) and these in turn fulfill the normalization conditions.

4.4.1 Details to *Learning the parameters of the probability model by EM*

In this section we will analyze the theoretical basis for learning the parameters $\mathbf{w}$ of the generative probability model $p(\mathbf{y}, k|\mathbf{w})$ given in Eq. (4.9) from a machine learning perspective. In contrast to the intuitive explanation of the Results section which was based on Expectation Maximization we will now derive an implicit analytical solution for a (locally) optimal weight vector $\mathbf{w}$, and rewrite this solution in terms of log probabilities. We will later use this derivation in order to show that the stochastic online learning rule provably converges towards this solution.

For an exact definition of the learning problem, we assume that the input is given by a stream of vectors $\mathbf{y}$, in which every $\mathbf{y}$ is drawn independently from the input distribution $p^*(\mathbf{y})$. In principle, this stream of $\mathbf{y}$'s corresponds to the samples $\mathbf{y}(t_1), \mathbf{y}(t_2), \ldots$ that are observed at the spike times $t_1, t_2, \ldots$ of the circuit. However, in order to simplify the proofs in this and subsequent sections, we will neglect any possible temporal correlation between successive samples.

The learning task is to find parameter values $\mathbf{w}$, such that the marginal $p(\mathbf{y}|\mathbf{w})$ of the model distribution $p(\mathbf{y}, k|\mathbf{w})$ approximates the actual input distribution $p^*(\mathbf{y})$ as accurately as possible. This is equivalent to minimizing the Kullback-Leibler divergence between the two distributions:

$$\begin{aligned} \mathrm{KL}(p^*(\mathbf{y})||p(\mathbf{y}|\mathbf{w})) &= \sum_{\mathbf{y}} p^*(\mathbf{y}) \log \frac{p^*(\mathbf{y})}{p(\mathbf{y}|\mathbf{w})} \\ &= -\mathrm{H}_{p^*}(\mathbf{y}) - \mathrm{E}_{p^*}[\log p(\mathbf{y}|\mathbf{w})] \quad , \end{aligned} \tag{4.29}$$

where $\mathrm{H}_{p^*}(\mathbf{y})$ is the (constant) entropy of the input distribution $p^*(\mathbf{y})$, and $\mathrm{E}_{p^*}[\cdot]$ denotes the expectation over $\mathbf{y}$, according to the distribution $p^*(\mathbf{y})$. Since $\mathrm{H}_{p^*}(\mathbf{y})$ is constant, minimizing the right hand side of Eq. (4.29) is equivalent to maximizing the expected log likelihood $L(\mathbf{w}) = \mathrm{E}_{p^*}[\log p(\mathbf{y}|\mathbf{w})]$.

There are many different parametrizations $\mathbf{w}$ that define identical generative distributions $p(\mathbf{y}, k|\mathbf{w})$ in Eq. (4.24). There is, however, exactly one $\mathbf{w}'$ in this sub-manifold of the weight space that fulfills the normalization conditions in Eq. (4.10).

We thus redefine the goal of learning more precisely as the constrained maximization problem

$$\max \quad L(\mathbf{w}) \tag{4.30}$$

$$\text{subject to} \quad \sum_{k=1}^{K} e^{w_{k0}} = 1 \quad \text{and} \quad \sum_{i \in G_j} e^{w_{ki}} = 1 \quad \text{for all } k, j \quad . \tag{4.31}$$

This maximization problem never has a unique solution $\mathbf{w}$, because any permutation of the values of k and their corresponding weights leads to different joint distributions $p(\mathbf{y}, k|\mathbf{w})$, all of them having identical marginals $p(\mathbf{y}|\mathbf{w})$. The local maxima of Eq. (4.30) can be found using the Lagrange multiplier method.

Note that we do at no time enforce normalization of $\mathbf{w}$ during the learning process, nor do we require normalized initialization of $\mathbf{w}$. Instead, we will show that the learning rule in Eq. (4.5,4.7) automatically drives $\mathbf{w}$ towards a local maximum, in which the normalization conditions are fulfilled.

Under the constraints in Eq. (4.31) the normalization constant Z in Eq. (4.21) equals 1, thus $L(\mathbf{w})$ simplifies to $\mathrm{E}_{p^*}[\log \sum_{k=1}^{K} e^{u_k}]$ - with $u_k = w_{k0} + \sum_{i=1}^{n} w_{ki} \cdot y_i$ - and we can define a Lagrangian function $\tilde{L}(\mathbf{w}, \boldsymbol{\lambda})$ for the maximization problem in Eq. (4.30,4.31) by

$$\tilde{L}(\mathbf{w}, \lambda) = \mathrm{E}_{p^*}[\log \sum_{k=1}^{K} e^{u_k}] - \lambda_0 \left(1 - \sum_{k=1}^{K} e^{w_{k0}}\right) - \sum_{k=1}^{K} \sum_{j=1}^{m} \lambda_{kj} \left(1 - \sum_{i \in G_j} e^{w_{ki}}\right) \quad . \tag{4.32}$$

Setting the derivatives to zero we arrive at the following set of equations in $\mathbf{w}$ and $\boldsymbol{\lambda}$:

$$\forall k: \qquad \frac{\partial \tilde{L}}{\partial w_{k0}} = \mathrm{E}_{p^*}[\frac{e^{u_k}}{\sum_{l=1}^{K} e^{u_l}}] - \lambda_0 e^{w_{k0}} = 0 \tag{4.33}$$

$$\forall k, i: \qquad \frac{\partial \tilde{L}}{\partial w_{ki}} = \mathrm{E}_{p^*}[y_i \frac{e^{u_k}}{\sum_{l=1}^{K} e^{u_l}}] - \lambda_{kj} e^{w_{ki}} = 0. \tag{4.34}$$

Summing over those equations that have the same multiplier λ_{kj} or λ_0, resp., leads to

$$\sum_{k=1}^{K} \frac{\partial \tilde{L}}{\partial w_{k0}} = \sum_{k=1}^{K} \mathrm{E}_{p^*}[p(k|\mathbf{y},\mathbf{w})] - \lambda_0 \sum_{k=1}^{K} e^{w_{k0}} = 0 \tag{4.35}$$

$$\forall k,j: \quad \sum_{i\in G_j} \frac{\partial \tilde{L}}{\partial w_{ki}} = \sum_{i\in G_j} \mathrm{E}_{p^*}[y_i\, p(k|\mathbf{y},\mathbf{w})] - \lambda_{kj} \sum_{i\in G_j} e^{w_{ki}} = 0 \quad , \tag{4.36}$$

where $p(k|\mathbf{y},\mathbf{w})$ is the shorthand notation for the equivalent expression $\frac{e^{u_k}}{\sum_{l=1}^{K} e^{u_l}}$. The identity $\sum_{k=1}^{K} \mathrm{E}_{p^*}[p(k|\mathbf{y},\mathbf{w})] = 1$, the identity $\sum_{i\in G_j} \mathrm{E}_{p^*}[y_i\, p(k|\mathbf{y},\mathbf{w})] = \mathrm{E}_{p^*}[p(k|\mathbf{y},\mathbf{w}) \sum_{i\in G_j} y_i]$ the fact that $\sum_{i\in G_j} y_i = 1$, which follows from the definition of population encoding, and the constraints in Eq. (4.31) are used in order to derive the explicit solution for the Lagrange multipliers

$$\lambda_0 = 1 \qquad \text{and} \qquad \forall k,j:\ \lambda_{kj} = \mathrm{E}_{p^*}[p(k|\mathbf{y},\mathbf{w})] \quad , \tag{4.37}$$

in dependence of $\mathbf{w}$. We insert this solution for $\boldsymbol{\lambda}$ into the gradient Eq. (4.33,4.34) and get

$$\begin{aligned} \mathrm{E}_{p^*}[p(k|\mathbf{y},\mathbf{w})] - e^{w_{k0}} &= 0 \\ \mathrm{E}_{p^*}[y_i\, p(k|\mathbf{y},\mathbf{w})] - \mathrm{E}_{p^*}[p(k|\mathbf{y},\mathbf{w})] e^{w_{ki}} &= 0 \quad , \end{aligned} \tag{4.38}$$

from which we derive an implicit solution for $\mathbf{w}$:

$$\begin{aligned} w_{k0} &= \log \mathrm{E}_{p^*}[p(k|\mathbf{y},\mathbf{w})] \\ w_{ki} &= \log \frac{\mathrm{E}_{p^*}[y_i\, p(k|\mathbf{y},\mathbf{w})]}{\mathrm{E}_{p^*}[p(k|\mathbf{y},\mathbf{w})]}. \end{aligned} \tag{4.39}$$

It is easily verified that all fixed points of this implicit solution satisfy the normalization constraints:

$$\sum_{k=1}^{K} e^{w_{k0}} = \sum_{k=1}^{K} \mathrm{E}_{p^*}[p(k|\mathbf{y},\mathbf{w})] = \mathrm{E}_{p^*}[\sum_{k=1}^{K} p(k|\mathbf{y},\mathbf{w})] = 1 \tag{4.40}$$

$$\sum_{i\in G_j} e^{w_{ki}} = \sum_{i\in G_j} \frac{\mathrm{E}_{p^*}[y_i\, p(k|\mathbf{y},\mathbf{w})]}{\mathrm{E}_{p^*}[p(k|\mathbf{y},\mathbf{w})]} = \frac{\mathrm{E}_{p^*}[p(k|\mathbf{y},\mathbf{w}) \sum_{i\in G_j} y_i]}{\mathrm{E}_{p^*}[p(k|\mathbf{y},\mathbf{w})]} = 1 \quad . \tag{4.41}$$

Finally, in order to simplify the notation we use the augmented input distribution

$p^*_{\mathbf{w}}(\mathbf{y}, \mathbf{z})$. The expectations in Eq. (4.39) nicely evaluate to

$$\mathrm{E}_{p^*}[p(z_k{=}1|\mathbf{y}, \mathbf{w})] = \sum_{\mathbf{y}} p^*(\mathbf{y}) p(z_k{=}1|\mathbf{y}, \mathbf{w}) = \sum_{\mathbf{y}} p^*_{\mathbf{w}}(\mathbf{y}, z_k{=}1) = \tag{4.42}$$

$$= p^*_{\mathbf{w}}(z_k{=}1) \quad \text{and} \tag{4.43}$$

$$\mathrm{E}_{p^*}[y_i\, p(z_k{=}1|\mathbf{y}, \mathbf{w})] = \sum_{\mathbf{y}} p^*(\mathbf{y})\, y_i\, p(z_k{=}1|\mathbf{y}, \mathbf{w}) = \sum_{\mathbf{y}} y_i\, p^*_{\mathbf{w}}(\mathbf{y}, z_k{=}1) = \tag{4.44}$$

$$= p^*_{\mathbf{w}}(y_i{=}1, z_k{=}1) \quad , \tag{4.45}$$

which allows us to rewrite the implicit solution in a very intuitive form as:

$$w_{k0} = \log p^*_{\mathbf{w}}(z_k = 1) \qquad\qquad w_{ki} = \log p^*_{\mathbf{w}}(y_i = 1|z_k = 1) \quad . \tag{4.46}$$

Any weight vector $\mathbf{w}$ that fulfills Eq. (4.46) is either a (local) maximum, a saddle point or a (local) minimum of the log likelihood function L under the normalization constraints.

An obvious numerical approach to solve this fixed point equation is the repeated application of Eq. (4.39). According to the derivations in the Results section this corresponds exactly to the Expectation Maximization algorithm. But every single iteration asks for the evaluation of expectations with respect to the input distribution $p^*(\mathbf{y})$, which theoretically requires infinite time in an online learning setup.

4.4.2 Details to *Spike-based Expectation Maximization*

We derive the update rule in Eq. (4.5) from the statistical perspective that each weight can be interpreted as $w_{ki} = \log \frac{a_{ki}}{N_{ki}}$, where a_{ki} and N_{ki} correspond to counters of the events $\langle y_i = 1, z_k = 1\rangle$ and $\langle z_k = 1\rangle$. Every new event $\langle y_i, z_k\rangle$ leads to a weight update

$$w_{ki}^{\text{new}} = \log \frac{a_{ki} + y_i z_k}{N_{ki} + z_k} = \tag{4.47}$$

$$= \log \frac{a_{ki}}{N_{ki}} (1 + \frac{1}{N_{ki}} \frac{N_{ki}}{a_i} y_i z_k)(1 + \frac{1}{N_{ki}} z_k)^{-1} \tag{4.48}$$

$$= w_{ki} + \log(1 + \frac{1}{N_{ki}} e^{-w_{ki}} y_i z_k) - \log(1 + \frac{1}{N_{ki}} z_k) \tag{4.49}$$

$$\approx w_{ki} + \frac{1}{N_{ki}} z_k (e^{-w_{ki}} y_i - 1) \quad , \tag{4.50}$$

where the log-function is linearly approximated around 1 as $\log(1 + x) \approx x$. The factor $\frac{1}{N_{ki}}$ is understood as learning rate η_{ki} in the additive update rule $w_{ki}^{\text{new}} = w_{ki} + \eta_{ki} \Delta w_{ki}$.

If $z_k = 0$, i.e. if there is no postsynaptic spike, the update $\Delta w_{ki} = 0$. In the case of a postsynaptic spike, i.e., $z_k = 1$, the update $\Delta w_{ki} = 0$ decomposes in the two cases $y_i = 1$ and $y_i = 0$ as it is stated explicit in Eq. (4.5).

As a side note, we observe that by viewing our STDP rule as an approximation to counting statistics, the learning rate $\eta_{ki} = \frac{1}{N_{ki}}$ can be understood as the inverse of the equivalent sample size from which the statistics was gathered. If the above rule is used with a small constant learning rate we will get a close approximation to an exponentially decaying average. If the learning rate decays like $\frac{1}{N_{ki}}$ we will get an approximation to an online updated average, where all samples are equally weighted. We will come back to a regulation mechanism for the learning rate in the section 'Variance Tracking'.

4.4.3 Details to *Proof of convergence*

In this section we give the proof of Theorem 1. Formally, we define the sequences $\mathbf{w}^{(t)}$, $\Delta\mathbf{w}^{(t)}$, $\mathbf{y}^{(t)}$, $\mathbf{z}^{(t)}$ and $\eta^{(t)}$ for $t = 0, 1, \ldots, \infty$: For all t we assume that $\mathbf{y}^{(t)}$ is drawn independently from $p^*(\mathbf{y})$. The value of $\mathbf{z}^{(t)}$ is drawn from the posterior distribution of the model $p(\mathbf{z}|\mathbf{y}^{(t)}, \mathbf{w}^{(t)})$ (see Eq. (4.11)), given the input $\mathbf{y}^{(t)}$ and the current model parameters $\mathbf{w}^{(t)}$. The weight updates $\Delta w_{ki}^{(t)}$, and $\Delta w_{k0}^{(t)}$, are calculated according to Eq. (4.5) and (4.7) with $c = 1$. The sequence of weight vectors $\mathbf{w}^{(t)}$ is determined by the randomly initialized vector $\mathbf{w}^{(0)}$, and by the iteration equation

$$\mathbf{w}^{(t+1)} = \Pi\left(\mathbf{w}^{(t)} + \eta^{(t)}\Delta\mathbf{w}^{(t)}\right) \quad . \tag{4.51}$$

The projection function Π represents a coordinate-wise clipping of $\mathbf{w}^{(t+1)}$ to a hyper-rectangle B such that

$$-w_{\min} \leq w_{ki}^{(t+1)} \leq 0 \qquad \text{and} \qquad -w_{\min} \leq w_{k0}^{(t+1)} \leq 0 \quad . \tag{4.52}$$

The bound $w_{\min}$ is assumed to be chosen so that all (finite) maxima of L are inside of B. For the sequence of learning rates $\eta^{(t)}$ we assume that

$$\sum_{t=1}^{\infty} \eta^{(t)} = \infty \qquad \text{and} \qquad \sum_{t=1}^{\infty} (\eta^{(t)})^2 < \infty \quad . \tag{4.53}$$

Under these assumptions we can now restate the theorem formally:

Theorem 1: *The sequence $\mathbf{w}^{(t)}$ converges with probability 1 to the set S_B of all points within the hyper-rectangle B that fulfill the equilibrium conditions in Eq. (4.6). The*

stable convergence points among S_B are the (local) maxima of L, subject to the normalization constraints in Eq. (4.10).

The iterative application of the learning rule in Eq. (4.5) and (4.7) is indeed a stochastic approximation algorithm for learning a (locally) optimal parameter vector $\mathbf{w}$. We resort to the theory of stochastic approximation algorithms as presented in [131] and use the method of the "mean limit" ordinary differential equation (ODE). The goal is to show that the sequence of the weight vector $\mathbf{w}^{(t)}$ under the stochastic learning rule in Eq. (4.5) and (4.7) converges to one of the local maxima of Eq. (4.30) with probability one, i.e., the probability to observe a non-converging realization of this sequence is zero. The location of the local maximum to which a single sequence of $\mathbf{w}^{(t)}$ converges depends on the starting point $\mathbf{w}^{(0)}$ as well as on the concrete realization of the stochastic noise sequence. We will not discuss the effect of this stochasticity in more detail, except for stating that a stochastic approximation algorithm is usually less prone to get stuck in small local maxima than its deterministic version. The stochastic noise introduces perturbations that decrease slowly over time, which has an effect that is comparable to simulated annealing.

We will use the basic convergence theorem of [131] to establish the convergence of the sequence $\mathbf{w}^{(t)}$ to the limit set of the mean limit ODE. Then it remains to show that this limit set is identical to the desired set of all equilibrium points and thus, particularly, does not contain limit cycles.

Proof: In the notation of [131], the mean update of the stochastic algorithm in Eq. (4.51) is $\bar{g}(\mathbf{w}) = \mathrm{E}_{p^*_{\mathbf{w}}}[\Delta\mathbf{w}^{(t)}]$. The bounds B imply that $\mathrm{E}_{p^*_w}[||\Delta\mathbf{w}^{(t)}||] < \infty$ for all t and $\sup_t \mathrm{E}_{p^*_w}[(\Delta\mathbf{w}^{(t)})^2] < \infty$.

For any set A we define $F(A)$ as the positive limit set of the mean limit ODE $\dot{\mathbf{w}}(s) = \bar{g}(\mathbf{w}(s))$ for all initial conditions $\mathbf{w}(0) \in A$:

$$F(A) = \lim_{s\to\infty} \bigcup_{\mathbf{w}\in A} \left\{\mathbf{w}(s'), s' \geq s : \mathbf{w}(0) = \mathbf{w}\right\} \quad . \tag{4.54}$$

According to Theorem 3.1 in Chapter 5 of [131], the sequence $\mathbf{w}^{(t)}$ under the algorithm in Eq. (4.51) converges for all start conditions $\mathbf{w}^{(0)} \in B$ to the limit set $F(B)$ with probability one in the sense that

$$\lim_{t\to\infty} \min_{\mathbf{w}\in F(B)} \left|\mathbf{w}^{(t)} - \mathbf{w}\right| = 0 \quad . \tag{4.55}$$

We will now show that the limit set $F(B)$ of $\dot{\mathbf{w}} = \bar{g}(\mathbf{w})$ is identical to the set of stationary points $S_B = \{\mathbf{w} \in B : \bar{g}(\mathbf{w}) = 0\}$ and does not contain limit cycles. It is obvious that S_B is a subset of $F(B)$ since for all initial conditions $\mathbf{w}(0) \in S_B$ the trajectory of $\dot{\mathbf{w}}(s) = \bar{g}(\mathbf{w}(s))$ fulfills $\mathbf{w}(s) \equiv \mathbf{w}(0)$ for all s. Thus it remains to be shown that there are no other points in F_B (like e.g. limit cycles).

We split the argument into two parts. In the first part we will show that for $s \to \infty$ all trajectories of $\dot{\mathbf{w}}(s) = \bar{g}(\mathbf{w}(s))$ converge asymptotically to the manifold H defined by the normalization constraints 4.31. This leads to the conclusion that $F(B \setminus H) \subset F(B \cap H)$. In the second part we will show that all trajectories within H converge to the stationary points S_B, i.e., $F(B \cap H) = S_B$. Both parts together yield the desired result that S_B are the only limit points of the ODE $\dot{\mathbf{w}}(s) = \bar{g}(\mathbf{w}(s))$.

The first part we start by defining the set of functions $h_0(\mathbf{w})$ and $h_{kj}(\mathbf{w})$ for all k, j to represent the deviation of the current $\mathbf{w}$ from each of the normalization constraints 4.31, i.e.,

$$h_0(\mathbf{w}) = \sum_{k=1}^{K} e^{w_{k0}} - 1 \qquad\qquad h_{kj}(\mathbf{w}) = \sum_{i \in G_j} e^{w_{ki}} - 1 \quad . \tag{4.56}$$

The manifold H is the set of all points $\mathbf{w}$ where $h_0(\mathbf{w}) = 0$ and $h_{kj}(\mathbf{w}) = 0$ for all k, j. Furthermore, we calculate the gradient vectors $\frac{\partial h_0}{\partial \mathbf{w}}$ and $\frac{\partial h_{kj}}{\partial \mathbf{w}}$ for each of these functions with respect to the argument $\mathbf{w}$. Note that many entries of these gradient vectors are 0, since every single function $h_{kj}(\mathbf{w})$ and $h_0(\mathbf{w})$ only depends on a few entries of its argument $\mathbf{w}$. The nonzero entries of these gradients are

$$\frac{\partial h_0}{\partial w_{k0}} = e^{w_{k0}} \qquad\qquad \forall i \in G_j : \quad \frac{\partial h_{kj}}{\partial w_{ki}} = e^{w_{ki}} \quad . \tag{4.57}$$

We can now show that the trajectory of $\dot{\mathbf{w}}(s) = \bar{g}(\mathbf{w}(s))$ in any point $\mathbf{w}(s)$ always points in direction of decreasing absolute values for all deviations $h_0()$ and $h_{kj}()$:

$$\bar{g}(\mathbf{w}) \cdot \frac{\partial h_0}{\partial \mathbf{w}} = \sum_{k=1}^{K} (\mathrm{E}_{p^*}[z_k] e^{-w_{k0}} - 1)\, e^{w_{k0}} \quad = \quad 1 - \sum_{k=1}^{K} e^{w_{k0}} \tag{4.58}$$

$$= -h_0(\mathbf{w}) \tag{4.59}$$

$$\bar{g}(\mathbf{w}) \cdot \frac{\partial h_{kj}}{\partial \mathbf{w}} = \sum_{i \in G_j} (\mathrm{E}_{p^*}[y_i\, z_k] e^{-w_{ki}} - \mathrm{E}_{p^*}[z_k])\, e^{w_{ki}} \quad = \quad p(k|\mathbf{y}, \mathbf{w})(1 - \sum_{i \in G_j} e^{w_{ki}}) \tag{4.60}$$

$$= -p(k|\mathbf{y}, \mathbf{w})\, h_{kj}(\mathbf{w}) \tag{4.61}$$

This shows that $\lim_{s\to\infty} h_{kj}(\mathbf{w}(s)) = 0$ for all k, j and $\lim_{s\to\infty} h_0(\mathbf{w}(s)) = 0$. This implies that the limit set of all trajectories with initial conditions outside H is contained in H, or more formally $F(B \setminus H) \subseteq B \cap H$. Note that the continuity and the boundedness of $\bar{g}(\mathbf{w})$ on B implies $F(F(A)) = F(A)$ and $F(A_1) \subseteq F(A_2)$ if $A_1 \subseteq A_2$ for all $A, A_1, A_2 \subseteq B$. Therefore we can now conclude as the result of the first part

$$F(B \setminus H) = F(F(B \setminus H)) \subseteq F(B \cap H) \quad , \tag{4.62}$$

i.e. the limit set of all trajectories starting outside the manifold of normalized weights is contained in the limit set of all trajectories starting within the normalization constraints. The equations (4.61) also prove that any trajectory with initial condition $\mathbf{w}(0) \in H$ stays within H, since all components of $\bar{g}(\mathbf{w}(s))$ with directions orthogonal to the tangent space of H in $\mathbf{w}(s)$ are 0 for all s, thus $\bar{g}(\mathbf{w}(s))$ is in the tangent space H in $\mathbf{w}(s)$.

This immediately leads to the second part of the proof, which is based on the gradient $\frac{\partial \tilde{L}}{\partial \mathbf{w}}$ of the Lagrangian $\tilde{L}$ as given in Eq. (4.33, 4.34). For any $\mathbf{w} \in H$ let $P(\mathbf{w})$ be the linear projection matrix that orthogonally projects any vector $\mathbf{a}$ into the tangent space of H in $\mathbf{w}$. The projection $P(\mathbf{w}) \cdot \frac{\partial \tilde{L}(\mathbf{w})}{\partial \mathbf{w}}$ of the gradient of $\tilde{L}$ at any $\mathbf{w} \in H$ points towards the strongest increase of the value of the objective function L under the constraints of the normalization conditions. Thus, the value of L increases in the direction of any vector within the tangent space of H in $\mathbf{w}$ that has a positive scalar product with $P(\mathbf{w}) \cdot \frac{\partial \tilde{L}(\mathbf{w})}{\partial \mathbf{w}}$. As $\bar{g}(\mathbf{w})$ is a tangent vector of H in $\mathbf{w}$ for all $\mathbf{w} \in H$, the orthogonal component $\frac{\partial \tilde{L}(\mathbf{w})}{\partial \mathbf{w}} - P(\mathbf{w}) \cdot \frac{\partial \tilde{L}(\mathbf{w})}{\partial \mathbf{w}}$ of the gradient is orthogonal to $\bar{g}(\mathbf{w})$. Thus, the value of the scalar product with the projected gradient $\bar{g}(\mathbf{w}) \cdot (P(\mathbf{w}) \cdot \frac{\partial \tilde{L}(\mathbf{w})}{\partial \mathbf{w}})$ is identical to the value of the scalar product with the gradient itself $\bar{g}(\mathbf{w}) \cdot \frac{\partial \tilde{L}(\mathbf{w})}{\partial \mathbf{w}}$:

$$\bar{g}(\mathbf{w}) \cdot \frac{\partial \tilde{L}(\mathbf{w})}{\partial \mathbf{w}} = \sum_{k=1}^{K} \frac{\partial L}{\partial w_{k0}} (\mathrm{E}_{p^*_{\mathbf{w}}}[z_k] e^{-w_{k0}} - 1) + \sum_{i,k} \frac{\partial L}{\partial w_{ki}} (\mathrm{E}_{p^*_{\mathbf{w}}}[z_k y_i] e^{-w_{ki}} - \mathrm{E}_{p^*_{\mathbf{w}}}[z_k]) \tag{4.63}$$

$$= \sum_{k=1}^{K} e^{-w_{k0}} (\mathrm{E}_{p^*}[p(z = k|\mathbf{y}, \mathbf{w}) - e^{w_{k0}}])^2 + \\ + \sum_{i,k} e^{-w_{ki}} (\mathrm{E}_{p^*}[(y_i - e^{w_{ki}}) p(z = k|\mathbf{y}, \mathbf{w})])^2 \quad \geq 0 \quad ,$$

with equality if and only if $\bar{g}(\mathbf{w}) = 0$, which is equivalent to $\frac{\partial \tilde{L}(\mathbf{w})}{\partial \mathbf{w}} = 0$. This shows that all trajectories with initial condition $\mathbf{w}(0) \in H$ stay within H forever and converge to the set of stationary points S_H, i.e. $F(B \cap H) = S_H$. Combining the results of both parts as

$$F(B) = F(B \setminus H) \cup F(B \cap H) = F(B \cap H) = S_B \tag{4.64}$$

establishes the stochastic convergences of any sequence $\mathbf{w}^{(t)}$ to the set S_B with probability one.■

Weight offsets and positive weights

All weights w_{ki} in the theoretical model are logs of probabilities and therefore always have negative values. Through a simple transformation we can shift all weights into the positive range in order to be able to use positive weights only, which is the common assumption for excitatory connections in biologically inspired neural network models. We will now show that setting the parameter c in Eq. (4.5) different from 1 leads to a linear shift of the resulting weight values by $\log\ c$, without changing the functionality of the Spike-based EM algorithm.

Firstly, we observe that the application of the update rule in Eq. (4.5) with $c > 1$ on a shifted weight $w'_{ki} = w_{ki} + \log c$ is identical to the application of the update rule with $c = 1$ on the original weight w_{ki}, since

$$ce^{-(\underbrace{w_{ki} + \log c}_{w'_{ki}})} - 1 = e^{-w_{ki}} - 1 \quad . \tag{4.65}$$

Secondly, we see that the relative firing rate $q_k(t)$ of neuron z_k remains unchanged if all weights are subject to the same offset $\log\ c$, since

$$q_k(t) = \frac{e^{w_{k0} + \sum_{i=1}^n (w_{ki} + \log c) y_i}}{\sum_{k'=1}^K e^{w_{k'0} + \sum_{i=1}^n (w_{k'i} + \log c) y_i}} \tag{4.66}$$

$$= \frac{\left(e^{\log c \sum_{i=1}^n y_i}\right) e^{w_{k0} + \sum_{i=1}^n w_{ki} y_i}}{\sum_{k'=1}^K \left(e^{\log c \sum_{i=1}^n y_i}\right) e^{w_{k'0} + \sum_{i=1}^n w_{k'i} y_i}} \tag{4.67}$$

$$= \frac{e^{w_{k0} + \sum_{i=1}^n w_{ki} y_i}}{\sum_{k'=1}^K e^{w_{k'0} + \sum_{i=1}^n w_{k'i} y_i}} \tag{4.68}$$

In contrast, the overall firing rate $R(t)$ increases by the factor $e^{\log c \sum_{i=1}^n y_i}$. By our definition of the population coding for $\mathbf{y}$, this factor equals $e^{m \log\ c}$, where m is the

number of original input variables $\mathbf{x}$. An increase of the inhibitory signal $I(t)$ by $m \log\ c$ can therefore compensate the increase of overall firing rate. Using this shifted representation, a single excitatory synapse can take on values in the range $[0, \log\ c]$, corresponding to probabilities in the range $[\frac{1}{c}, 1]$.

Similarly the consideration holds valid that it is mathematically equivalent whether the depression of the excitability w_{k0} in Eq. (4.7) is modeled either as an effect of lateral spiking activity or as a constant decay, independent of the circuit activity. In the first case, w_{k0} converges to the relative spiking probability of the $k-$th neuron such that the sum of all w_{k0} is indeed 1 as described by our theory. In the second case, the w_{k0} really describe absolute firing rates in some time scale defined by the decay constant. In the logarithmic scale of w_{k0} this is nothing else than a constant offset and thus cancels down in Eq. (4.68).

Impact of missing input values

The proof of theorem 1 assumes that every sample $\mathbf{y}^{(t)}$ gathered online is a binary vector which contains exactly one entry with value 1 in every group G_j. This value indicates the value of the abstract variable x_j that is encoded by this group. As long as the spikes from the input neurons are closely enough in time, this condition will be fulfilled for every activation vector $\mathbf{y}(t)$. For the cases in which the value of the abstract variable x_j changes, the first spike from group G_j has to appear exactly at that point in time at which the rectangular EPSP for the previous value vanishes, i.e., σ ms after the last preceding spike.

We will now break up this strong restriction of the provable theory and analyze the results that are to be expected, if we allow for interspike intervals longer than σ. We interpret the resulting "gaps" in the information about the value of an input group as missing value in the sense of Bayesian inference.

We had already addressed the issue of such missing values, resulting from presynaptic neurons that do not spike within the integration time window of an output neuron z_k, in the discussion of Fig. 4.3.

A profound analysis of the correct handling of missing data in EM can be found in [74]. Their analysis implies that the correct learning action would be to leave all weights w_{ki} in the group G_j unchanged, if the value of the external variable x_j is missing, i.e., if all corresponding y_i's are 0. However, in this case the STDP rule in Eq. (4.5) reduces

these weights by η. This leads to a modification of the analysis of the equilibrium condition (4.28):

$$\begin{aligned} \mathrm{E}[\Delta w_{ki}] = 0 \;\Leftrightarrow\; & (1-r)\left(p^*(y_i{=}1|z_k{=}1)\eta(e^{-w_{ki}}-1) - p^*(y_i{=}0|z_k{=}1)\eta\right) - r\eta = 0 \\ \Leftrightarrow\quad & w_{ki} = \log p^*(y_i{=}1|z_k{=}1) + \log(1-r) \quad , \end{aligned} \tag{4.69}$$

where r is the probability that i belongs to a group G_j in which the value of x_j is unknown. We assume that the probability for such a missing value event is independent of the (true) value of the abstract variable x_j and we assume further that the probability of such missing value events is the same for all groups G_j and thus conclude that this offset of $\log(1-r)$ is expected to be the same for all weights. It can easily be verified, that such an offset does not change the resulting probabilities of the competition in the inference according to Eq. (4.68).

4.4.4 Adaptive learning rates with Variance Tracking

In our experiments we used an adaptation of the variance tracking heuristic from [158] for an adaptive control of learning rates. If we assume that the consecutive values of the weights represent independent samples of their true stochastic distribution at the current learning rate, then this observed distribution is the log of a beta-distribution defined by the parameters a_{ki} and N_{ki} that were used in Eq. (4.50) to define the update of w_{ki} from sufficient statistics. Analytically (see supplement) this distribution has the first and second moments

$$\mathrm{E}[w_{ki}] \approx \log\frac{a_{ki}}{N_i} \qquad \text{and} \qquad \mathrm{E}[w_{ki}^2] \approx \mathrm{E}[w_{ki}]^2 + \frac{1}{a_{ki}} + \frac{1}{N_i} \quad . \tag{4.70}$$

From the first equation we estimate $\frac{1}{a_{ki}} = \frac{e^{-\mathrm{E}[w]}}{N_i}$. This leads to a heuristic estimate for the (inverse of the) current sample size based on the empirically observed variance $\mathrm{E}[w_{ki}^2] - \mathrm{E}[w_{ki}]^2$:

$$\eta_{ki}^{new} = \frac{1}{N_i} = \frac{\mathrm{E}[w_{ki}^2] - \mathrm{E}[w_{ki}]^2}{e^{-\mathrm{E}[w_{ki}]} + 1}. \tag{4.71}$$

The empirical estimates of these first two moments can be gathered online by exponentially decaying averages using the same learning rate η_{ki}. Even though the assumption of independent samples for the estimates of the moments is not met, one can argue about two cases: In case of a stationary evolution of the weight, the strong dependence

of consecutive samples typically leads to an underestimation of the variance. This in turn leads to a decrease of the learning rate which is the desired effect of a stationary evolution. In case of a directed evolution of the weight the variance will at least indicate the amount of the current gradient of the evolution despite the strong dependence and thus keep the learning rate high enough to support fast convergence towards the asymptote of the gradient.

An adaptive learning rate such as in Eq. (4.71) facilitates a spontaneous reorganization of the internal models encoded by the weight vectors of the output neurons z_k in case that the input distribution $p^*(\mathbf{y})$ changes (see Fig. S1 in Text S1).

4.4.5 Details to *Role of the Inhibition*

Biased sampling problem

In this section we analyze the influence of the instantaneous output firing rate $R(t)$ of the learning circuit and derive the analytical result that the output rate $R(t)$ plays the role of a multiplicative weighting of samples during learning. We show how a theoretically optimal inhibition signal can compensate this effect and describe how this compensation is approximated in our experiments.

We start with the assumption that the input signal $\mathbf{y}(t)$ can be described by some stationary stochastic process. An empirical estimate of its stationary distribution can be obtained by measuring the relative duration of presentation of every different discrete value $\mathbf{y}$ in a time window of length T. The accuracy of this empirical estimate of the input distribution can be increased by using a longer time window T, such that in the limit of an infinitely large time window the estimate will converge to the true stationary input distribution of $\mathbf{y}$, denoted by $p^*(\mathbf{y})$:

$$p^*(\mathbf{y}) = \lim_{T\to\infty} \frac{1}{T} \int_0^T \delta(\mathbf{y} - \mathbf{y}(t))\, dt \quad , \tag{4.72}$$

where δ is a vectorized version of the Kronecker Delta with $\delta(\mathbf{0}) = 1$ and $\delta(\mathbf{x}) = 0$, if $\mathbf{x} \neq \mathbf{0}$.

However, even though the WTA-circuit receives this time-continuous input stream $\mathbf{y}(t)$, the spike-triggered STDP rule in Eq. (4.5) and (4.7) updates the model parameters - i.e. the synaptic weights - only at those time points where one of the output neurons

spikes. We denote by $p^*_S(\mathbf{y})$ the (empirical) distribution that is obtained from the observations of $\mathbf{y}(t)$ at the first S spike events $t^f_1, t^f_2, \ldots, t^f_S$:

$$p^*_S(\mathbf{y}) = \frac{1}{S}\sum_{s=1}^{S}\delta(\mathbf{y}-\mathbf{y}(t^f_s)) \quad . \tag{4.73}$$

The distribution $p^*_S(\mathbf{y})$ that is seen by the learning rule in Eq. (4.5) depends not only on the time-continuous input stream $\mathbf{y}(t)$, but also on the concrete spike times t^f_s of the circuit. The output spikes thus serve as trigger events at which the continuous input signal is *sampled*.

The spike times $t^f_1, t^f_2, \ldots, t^f_S$ and the total number of spikes S of the whole circuit within a time window of length T are distributed according to an inhomogeneous Poisson process with the instantaneous rate $R(t)$. For any stochastic realization of S and t^f_s in the time interval 0 to T, we can derive the expectation of the function $p^*_S(\mathbf{y})$ by taking the limit for $T\to\infty$ and call this the expected empirical distribution $p^*_{R(t)}(\mathbf{y})$. Thus

$$p^*_{R(t)}(\mathbf{y}) = \lim_{T\to\infty} \mathrm{E}_{S,t^f_1,\ldots,t^f_S}\left[\frac{1}{S}\sum_{s=1}^{S}\delta(\mathbf{y}-\mathbf{y}(t^f_s))\right] \tag{4.74}$$

$$= \lim_{T\to\infty} \mathrm{E}_S\left[\mathrm{E}_{t^f_1,\ldots,t^f_S}\left[\frac{1}{S}\sum_{s=1}^{S}\delta(\mathbf{y}-\mathbf{y}(t^f_s))\,\middle|\, S\right]\right] \quad , \tag{4.75}$$

where we divided the expectation into two parts. Firstly we take the expectation over the total number S of spikes, secondly we take the expectation over the spike times $t_1, \ldots, t_S$, given S. We now make use of the fact that for any inhomogeneous Poisson process $R(t)$, conditioned on the total number of events S within a certain time window T, the event times $t^f_1, \ldots, t^f_S$ are distributed as order statistics of S unordered independent samples $t'^f_1, \ldots, t'^f_S$ from the probability density $\frac{R(t')}{\int_0^T R(t)dt}$. The expectation $\mathrm{E}_{t'^f_s}[f(t'^f_s)|S]$ over an arbitrary function $f()$ is the integral $\int_0^T \frac{R(t')}{\int_0^T R(t)dt} f(t')dt'$, independent of the event number s, thus

$$p^*_{R(t)}(\mathbf{y}) = \lim_{T\to\infty} \mathrm{E}_S\left[\frac{1}{S}\sum_{s=1}^{S}\mathrm{E}_{t'^f_s}\left[\delta(\mathbf{y}-\mathbf{y}(t'^f_s))\,\middle|\, S\right]\right] \tag{4.76}$$

$$= \lim_{T\to\infty} \mathrm{E}_S\left[\frac{1}{S}S\int_0^T \frac{R(t')}{\int_0^T R(t)dt}\delta(\mathbf{y}-\mathbf{y}(t'))dt'\right] \tag{4.77}$$

$$= \lim_{T\to\infty} \mathrm{E}_S\left[\int_0^T \frac{R(t')}{\int_0^T R(t)dt}\delta(\mathbf{y}-\mathbf{y}(t'))dt'\right] \quad . \tag{4.78}$$

Since the remaining term within the expectation operator E_S is independent of S we obtain the final result

$$p^*_{R(t)}(\mathbf{y}) = \lim_{T\to\infty} \frac{1}{\int_0^T R(t)dt} \int_0^T R(t)\delta(\mathbf{y} - \mathbf{y}(t))dt \quad . \tag{4.79}$$

This shows that the output rate $R(t)$ acts as a multiplicative weighting of the contribution of the current input $\mathbf{y}(t)$ to the expected empirical distribution $p^*_{R(t)}(\mathbf{y})$, which is learned in the limit of $t \to \infty$ by the simple STDP rule in Eq. (4.5) and (4.7).

It turns out that the condition of a constant rate $R(t)$ is by far stronger than necessary. In fact, it is easy to see from a comparison of Eq. (4.72) and Eq. (4.79), that $p^*_{R(t)}(\mathbf{y}) = p^*(\mathbf{y})$ for all values of $\mathbf{y}$ if and only if the relative weight for the input value $\mathbf{y}$, which is $\frac{\int_0^T R(t)\delta(\mathbf{y}-\mathbf{y}(t))\,dt}{\int_0^T \delta(\mathbf{y}-\mathbf{y}(t))\,dt}$, is independent of $\mathbf{y}$ in the limit $T \to \infty$. This is certainly true if $R(t)$ and $\mathbf{y}(t)$ are stochastically independent, i.e. $R(t)$ is not correlated to the occurrence of any specific value of $\mathbf{y}$.

Inhibition Model in Computer Simulations

In our computer simulation the inhibition is implemented by adding a strongly negative impulse to the membrane potential of all z-neurons whenever one of them fires, which decays with a time constant of 5 ms back to its resting value. In addition, a noise term $v(t)$ is added to the membrane potential $u_k(t)$ that models background synaptic inputs through an Ornstein-Uhlenbeck (OU) process (as proposed in [50] for modeling in-vivo conditions) and causes stochastic firing. For each experiment, all parameters for the inhibition model are listed in "Simulation Parameters" in the Supplementary Material.

4.4.6 Details to *Continuous-Time Interpretation with Realistically Shaped EPSPs*

Let the external input vector $\mathbf{x}$ consist of multiple discrete-valued functions in time $x_j(t)$, and let us assume that for every input x_j there exists an independent Poisson sampling process with rate r_j which generates spike times for the group of neurons y_i with $i \in G_j$. At every spike time t_j^f there is exactly one neuron in the group that fires a spike, and this is the neuron that is associated with the value $x_j(t_j^f)$. First, we analyze additive step-function EPSPs, i.e. the postsynaptic activation $\tilde{y}_i(t)$ is given by the convolution in Eq. (4.17) where K is a step-function kernel with $K(t) = 1$ for $0 < t < \sigma$

for a fixed EPSP-duration σ and $K(t) = 0$ otherwise. In order to understand the resulting distribution $q_k(t)$ in Eq. (4.4) as Bayesian inference we extend our underlying generative probabilistic model $p(\mathbf{x}, k|\theta)$ such that it contains multiple instances of the variable vector $\mathbf{x}$, called $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(L)}$, where L is the total number of spikes from all input neurons y_i within the time window $[t - \sigma, t]$. We can see every spike as a single event in continuous time. The full probabilistic model is defined as

$$p(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(L)}, k|\mathbf{w}) = p(k|\mathbf{w}) \prod_{l=1}^{L} p(\mathbf{x}^{(l)}|k, \mathbf{w}) \quad , \tag{4.80}$$

which defines that the multiple instances are modeled as being conditionally independent of each other, given k. Let the vectors $\hat{\mathbf{y}}^{(l)}$ describe the corresponding spike "patterns" in which every binary vector $\hat{\mathbf{y}}^{(l)}$ has exactly one 1 entry $\hat{y}^{(l)}_{i^{(l)}} = 1$. All other values are zero, thus it represents exactly one evidence for $\mathbf{x}^{(l)}$, i.e. $x^{(l)}_j = v(i^{(l)})$, with j, s.t. $i^{(l)} \in G_j$, according to the decoding in Eq. (4.8).

Due to the conditional independences in the probabilistic model every such evidence, i.e. every spike, contributes one factor $p(\hat{y}^{(l)}_i = 1|k, \mathbf{w})$ to the likelihood term in the inference of the hidden node k. The inference is expressed as

$$p(k|\hat{\mathbf{y}}^{(1)}, \ldots, \hat{\mathbf{y}}^{(L)}, \mathbf{w}) = \frac{\overbrace{e^{w_{k0}}}^{\text{prior } p(k|\mathbf{w})} \cdot \overbrace{\prod_{i=1}^{n} (e^{w_{ki}})^{\sum_{l=1}^{L} \hat{y}^{(l)}_i}}^{\text{likelihood } p(\hat{\mathbf{y}}^{(1)}, \ldots, \hat{\mathbf{y}}^{(L)}|k, \mathbf{w})}}{\underbrace{\sum_{k'=1}^{K} e^{w_{k'0}} \prod_{i=1}^{n} (e^{w_{k'i}})^{\sum_{l=1}^{L} \hat{y}^{(l)}_i}}_{p(\hat{\mathbf{y}}^{(1)}, \ldots, \hat{\mathbf{y}}^{(L)}|\mathbf{w})}} \quad . \tag{4.81}$$

The identity $\tilde{y}_i(t) = \sum_{l=1}^{L} \hat{y}^{(l)}_i$ reveals that the above posterior distribution is realized by the relative spike probability $q_k(t)$ of the network model according to Eq. (4.4), where $\tilde{\mathbf{y}}(t)$ replaces $\mathbf{y}(t)$ in the computation of the membrane potential $u_k(t)$. Due to the step function $K(t)$ the result of the convolution in $\tilde{y}_i(t)$ equals the number of spikes within the time window $[t - \sigma, t]$ from neuron y_i. The factor $e^{w_{ki}}$, which has the meaning $p(y_i = 1|k, \mathbf{w})$ in the network model, is multiplied $\tilde{y}_i(t)$ times to the likelihood.

The above discrete probabilistic model gives an interpretation only for integer values of $\tilde{y}_i(t)$, i.e. for functions K such that $K(t)$ is 0 or any positive integer at any time t. For an interpretation of arbitrarily shaped EPSPs $K(t)$ - especially for continuously

decaying functions - in the context of our probabilistic model, we now extend this weighting mechanism from integer valued weights to real valued weights by a linear interpolation of the likelihood in the log-space.

The obvious restrictions on the EPSP function $K(t)$ are that it is non-negative, zero for $t < 0$, and $\int_0^\infty K(t)dt < \infty$, in order to avoid acausal or nondecaying behavior, and unboundedly growing postsynaptic potentials at constant input rates. We assume the normalization $\max K(t) = 1$. Let again $t^{(1)}, \ldots, t^{(l)}, \ldots$ be the times of the past spiking events and $i^{(1)}, \ldots, i^{(l)}, \ldots$ be the indices of the corresponding input neurons. The output distribution $q_k(t)$can be written as

$$q_k(t) = \frac{e^{w_{k0}} \prod_{l=1}^{\infty} (e^{w_{ki^{(l)}}})^{K(t-t^{(l)})}}{\sum_{k'=1}^{K} e^{w_{k0}} \prod_{l=1}^{\infty} (e^{w_{ki^{(l)}}})^{K(t-t^{(l)})}} \quad , \tag{4.82}$$

which nicely illustrates that every single past spike at time $t^{(l)}$ is seen as an evidence in the inference, but that evidence is weighted with a value $K(t - t^{(l)})$, which is between 0 and 1.

The analogous interpolation for continuous-valued input activations $\tilde{y}_i(t)$ yields the learning rule in Eq. (4.18), which is illustrated in Fig. 4.2 as the "Complex STDP rule" (blue dashed curve). The resulting shape of the LTP part of the STDP curve is determined by the EPSP shape defined by $K(t)$. The positive part of the update in Eq. (4.18) is weighted by the value of $\tilde{y}_i(t)$ at the time of firing the postsynaptic spike. Negative updates are performed if $\tilde{y}_i(t)$ is close to zero, which indicates that no presynaptic spikes were observed recently.

The proof of stochastic convergence does not explicitly assume that $\mathbf{y}^{(t)}$ is a binary vector, but is valid for any (positive) random variable vector $\tilde{\mathbf{y}}^{(t)}$ with finite variance. Further, the proof assumes the condition that in every group G_j the sum of the input activities $\tilde{y}_i^{(t)}$ is 1 at all times or at least at those points in time at which one z_k neuron of the WTA-circuit fires. The condition can be relaxed such that the sum per group does not have to be equal to 1 but to any arbitrary (positive) constant if the corresponding normalization constraint is adapted accordingly. Due to the decaying character of the EPSP shape, this sum will never stay constant, even for very regular input patterns. If we only assumed a constant average activation within a group, allowing for stochastic fluctuations around the target value, it turns out that this condition alone is not enough. We need to further assume that these stochastic fluctuations in the sum of every input

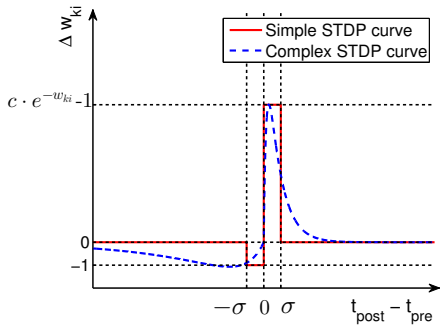

Figure 4.9: STDP learning curves with time-dependent LTD. Under the simple STDP model (red curve), weight-dependent LTP occurs only if the postsynaptic spike falls within a time window of length σ after the presynaptic spike, and LTD occurs in a time window of the same length, but for the opposite order of spikes. This can be extended to a more complex STDP rule (blue dashed curve), in which both LTP and LTD follow α-kernels with different time constants, typically with longer time-constants for LTD.

group G_j are stochastically independent of the circuit's response z_k. This assumption is intricate and may depend on the data and the learning progress itself, so it will usually not be exactly fulfilled. We can, however, argue that we are close to independence if at least the sum of activity in every group G_j is independent of the value of the underlying abstract variable x_j.

In our simulations we obtain the input activations $\tilde{y}_i(t)$ by simulating biologically realistic EPSPs at every synapse, using α-kernels with plausible time constants to model the contributions of single input spikes.

4.4.7 Details to *Spike-timing dependent LTD*

We formalize the presynaptic activity of neuron y_i *after* a postsynaptic spike at time t^f by ν_i, s.t. $\nu_i = 1$ if there is a spike from neuron y_i within the time window $[t^f, t^f + \sigma]$ and $\nu_i = 0$ otherwise. This trace is used purely for mathematical analysis, and cannot be known to the postsynaptic neuron at time t^f, since the future input activity is unknown. Mechanistically, however, ν_i can be implemented as a trace updated by postsynaptic firing, and utilized for plasticity at the time of presynaptic firing [203]. Let us now consider the STDP rule illustrated by the red curve in Fig. 4.9, where a depression of the synapse happens only if there is a presynaptic spike within the short time window of length σ *after* the postsynaptic spike, i.e. if $\nu_i = 1$. The application

of this STDP-rule in our neuronal circuit is equivalent to the circuit-spike triggered update rule

$$\Delta w_{ki} = z_k(c \cdot e^{-w_{ki}} y_i - d \cdot \nu_i) \tag{4.83}$$

which replaces Eq. (4.5). In analogy to Eq. (4.6) the equilibrium of this new update rule can be derived as

$$\mathrm{E}[\Delta w_{ki}] = 0 \Leftrightarrow p(y_i = 1, z_k = 1) c e^{-w_{ki}} - p(\nu_i = 1, z_k = 1) d = 0 \tag{4.84}$$

$$\Leftrightarrow w_{ki} = \log p^*(y_i = 1 | z_k = 1) - \log p^*(\nu_i = 1 | z_k = 1) + \log \frac{c}{d}, \tag{4.85}$$

under the assumption that y_i, ν_i and z_k are sampled from a stationary distribution $p^*(y_i, \nu_i, z_k)$. This shows that the synaptic weights can be interpreted as the log-likelihood ratio of the presynaptic neuron firing before instead of after the postsynaptic neuron. In other words, the neuron's synaptic weights learn the contrast between the current input pattern y_i that caused firing, and the following pattern of activity ν_i. Note that any factor c (for LTP) or d (for LTD) only leads to a constant offset of the weight which - under the assumption that the offset is the same for all synapses - can be neglected due to the WTA circuit (see Methods "Weight offsets and positive weights").

Similarly to our analysis for the standard SEM rule, we can derive a continuous-time interpretation of the timing-dependent LTD rule. As we did in Eq. (4.17), we can define

$$\tilde{y}_i(t) = \sum_f K_P(t - t_i^f) \qquad \tilde{\nu}_i(t) = \sum_f K_D(t - t_i^f), \tag{4.86}$$

where K_P is the same convolution kernel as in Eq. (4.17), and K_D is an arbitrary but time-inversed kernel, such that $K_D(t) = 0$ for positive t and $K_D(t) > 0$ for negative t. The value of ν_i thus reflects a time-discounted sum of presynaptic activity immediately after the postsynaptic spike.

The complex STDP rule from Fig. 4.2, which models LTD as a constant time-independent depression, can be seen as an extreme case of the spike-timing dependent LTD rule. If K_D is a step function with $K_D(t) = \frac{1}{\sigma}$ in the interval $[-\sigma, 0]$ and 0 everywhere else, then ν_i is just the average rate of presynaptic activity in the time interval $[t^f, t^f + \sigma]$ following a postsynaptic spike. In the limit of $\sigma \to \infty$ this is equivalent to the overall spiking rate of the neuron y_i, which is proportional to the

marginal $p(y_i)$ in the probabilistic model. Precisely, $\nu_i \to rp(y_i)$, where r is the base firing rate of an active input in our input encoding model. The equilibrium point of every weight w_{ki} becomes $\log p(y_i = 1|z_k = 1) - \log p(y_i)$, neglecting the offsets induced by the constants c,d and r. It is easy to see that the probabilistic interpretation of the neuronal model from Eq. (4.4) is invariant under the transformation $w'_{ki} = w_{ki} - \log p(y_i)$, since

$$q_k(t) = \frac{e^{w_{k0}+\sum_{i=1}^{n}(w_{ki}-\log p(y_i))y_i}}{\sum_{k'=1}^{K} e^{w_{k'0}+\sum_{i=1}^{n}(w_{k'i}-\log p(y_i))y_i}} \tag{4.87}$$

$$= \frac{\left(e^{\sum_{i=1}^{n} y_i \log p(y_i)}\right) e^{w_{k0}+\sum_{i=1}^{n} w_{ki}y_i}}{\sum_{k'=1}^{K} \left(e^{\sum_{i=1}^{n} y_i \log p(y_i)}\right) e^{w_{k'0}+\sum_{i=1}^{n} w_{k'i}y_i}} \tag{4.88}$$

$$= \frac{e^{w_{k0}+\sum_{i=1}^{n} w_{ki}y_i}}{\sum_{k'=1}^{K} e^{w_{k'0}+\sum_{i=1}^{n} w_{k'i}y_i}} \quad , \tag{4.89}$$

which proves that in our network model the complex STDP rule from Fig. 4.2 is equivalent to an offset-free STDP rule in the limit of an arbitrarily long window for LTD. In practice, of course, we can assume that the times between pre- and post-synaptic spikes are finite, and we have shown in Fig. 4.4 that as a result, very realistic shapes of STDP curves emerge at intermediate stimulation frequencies.

4.5 Supplement

4.5.1 Derivation of Variance tracking

For the derivation of Eq. (4.70), let q be a random variable distributed according to a Beta-Distribution with parameters a and b

$$p(q) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} p^{a-1}(1-p)^{b-1} \quad . \tag{4.90}$$

Let $w = \log q$, than w is distributed as follows:

$$p(w) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} (e^w)^a (1-e^w)^{b-1} \tag{4.91}$$

In order to calculate $\mathrm{E}[w]$ and $\mathrm{E}[w^2]$ we use the moment-generating function of $p(w)$

$$M_w(s) = \int_{-\infty}^{0} e^{sw} p(w) \mathrm{d}w = \tag{4.92}$$

$$= \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \int_{-\infty}^{0} (e^w)^{a+s} (1-e^w)^{b-1} \mathrm{d}w = \tag{4.93}$$

$$= \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \frac{\Gamma(a+s)\Gamma(b)}{\Gamma(a+b+s)} = \frac{\Gamma(a+b)\Gamma(a+s)}{\Gamma(a)\Gamma(a+b+s)} \quad . \tag{4.94}$$

The first and the second derivative of M_w read

$$M_w'(s) = \frac{\Gamma(a+b)\Gamma(a+s)}{\Gamma(a)\Gamma(a+b+s)} (\psi(a+s) - \psi(a+b)) \tag{4.95}$$

$$M_w''(s) = \frac{\Gamma(a+b)\Gamma(a+s)}{\Gamma(a)\Gamma(a+b+s)} \Big((\psi(a+s) - \psi(a+b))^2 + \psi_1(a+s) - \psi_1(a+b+s) \Big) \quad . \tag{4.96}$$

Since $\mathrm{E}[w] = M_w'(0)$ and $\mathrm{E}[w^2] = M_w''(0)$ we get

$$\mathrm{E}[w] = \psi(a) - \psi(a+b)$$
$$\mathrm{E}[w^2] = \mathrm{E}[w]^2 + \psi_1(a) - \psi_1(a+b)$$

which can be simplified using the approximations $\psi(x) \approx \log(x)$ and $\psi_1(x) \approx \frac{1}{x}$ to

$$\mathrm{E}[w] \approx \log \frac{a}{a+b} \qquad\qquad \mathrm{E}[w^2] \approx \frac{1}{a} + \frac{1}{a+b} \tag{4.97}$$

4.5.2 Adaptation to changing input distributions

In this computer experiment, 10 output neurons learned implicit generative models for images of handwritten digits from the MNIST database. The same procedure for encoding the images by spike trains as in Fig. 4.6 was used. Initially, only images representing the digits 0 and 3 were presented, and the WTA circuit learned accurate probabilistic models for these images. After 100 seconds of learning, the input distribution was changed, and a third class of inputs, images of handwritten digits 4, was introduced. Through the adaptive learning rate from Eq. (4.71), the z_k neurons spontaneously reorganized, and two output neurons changed their internal models to represent the new digit 4. In the end, an accurate generative model for all three types of input images was learned.

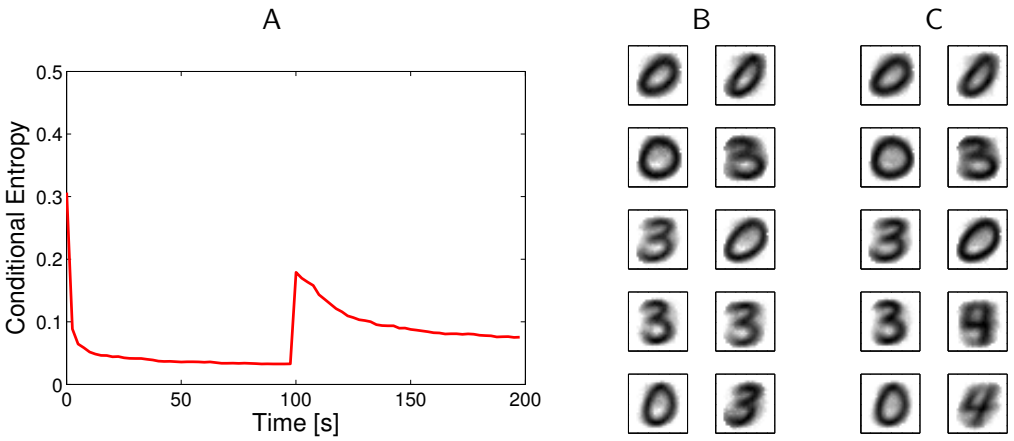

Figure 4.10: Spontaneous reorganization of the ensemble of internal models when the input distribution $p^*(\mathbf{y})$ changes. **A**: Time course of conditional entropy when after 100 s new, previously unseen samples of images of handwritten digits 4 were added to samples of handwritten digits 0 and 3. **B**: Weight vectors of the 10 output neurons after 100 s of learning (before the change of the input distribution). **C**: Spontaneous reorganization of these weight vectors after further 100 s. The weight vectors of two output neurons z_k have developed internal models for two ways of writing the (new) digit 4. Encoding of handwritten digits from MNIST by spike trains $\mathbf{y}$ is as in Fig. 4.6. The adaptive learning rate in Eq. (4.71) was used for this experiment.

4.5.3 Invariance to Time-Warping

4.5.4 Simulation Parameters

All simulations were carried out in MATLAB, with a simulation time step of 1 ms. The time constant of the OU process that modeled background synaptic inputs was set to 5 ms, its variance to 2500.

Simulations for Fig. 4.3:

Input generation:

For each input image pixels were drawn over a 28 x 28 array from one of 4 symmetrical Gaussians with $\sigma^2 = 10$ and centers at (14,8), (16,22), (9,15), (20,14), with maximal probability 0.3 for any pixel to be drawn (causing high variability of samples from the same Gaussian). In addition any pixel was drawn with probability 0.03 (added noise).

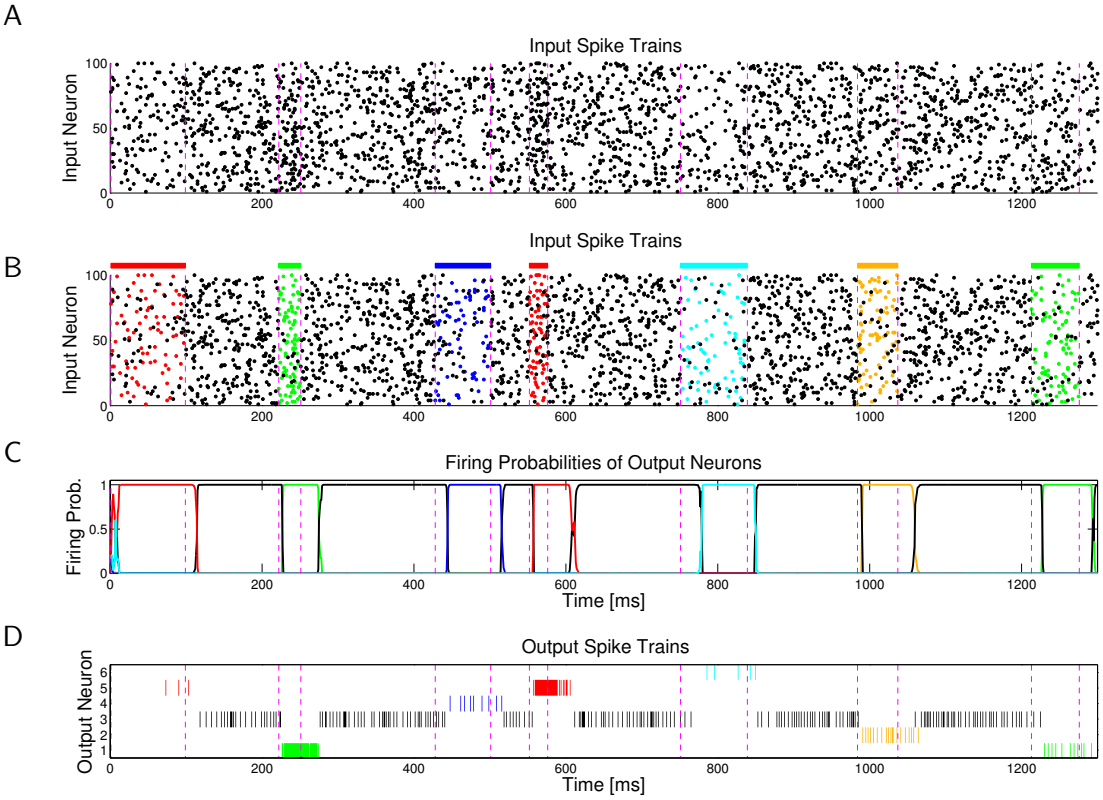

Figure 4.11: Generalization capability of the output neurons from Fig. 4.7 for time-warped variation of the input patterns. A, B: Another test input presented to the circuit from Fig. 4.7. The noise-embedded spike patterns are now compressed or stretched from 50 ms to a random length between 25 and 100 ms. Such time-warped versions of these patterns had never been presented during learning via STDP. **C, D**: Firing probabilities and spike outputs of the same 6 output neurons as in Fig. 4.7. They demonstrate that the emergent discrimination ability of these 6 output neurons automatically generalizes to time-warped input patterns (embedded into noise).

When an output neuron z_k fired, on average only 8.6% of the input neurons y_i had fired during the preceding 10 ms (the time window for potentiation according to the STDP rule in Eq. (4.5). Hence for over 90% of the pixels no spike was received within that time window from either one of the two neurons y_i that encoded the value of this pixel by population coding. The corresponding average activity level of all input synapses was at 0.182.

In the variation with superimposed background oscillations at 20 Hz the firing rates of input neurons y_i did not rise, but the average synaptic activity level at the time of an output spike rose to 0.215, an increase of around 18%. This leads to an increased learning rate.

The mean (offset) μ_{ou} of the OU-noise was set to 200, the initial value A_{inh} of lateral inhibition (caused by a firing of a z-neuron) was set to 3000, its resting value O_{inh} to 550. For the version with background oscillations (at 20 Hz) the amplitude of the oscillation was set to 500 (mean = 0), and the phase was shifted by 5 ms for the z-neurons, $A_{inh} = 3000$, $O_{inh} = 650$.

Simulation for Fig. 4.5:

$\mu_{ou} = 1000, A_{inh} = 3000, O_{inh} = 550.$

Simulation for Fig. 4.4:

In Figs. 4.4 A-C pre- and post-synaptic neurons were forced to fire at frequencies of $1, 20$, and 40 Hz with different time delays. The weight was kept fixed at $w = 3.5$ for $c = e^{-5}$, and the learning rate was kept fixed at $\eta = 0.5$. For Fig. 4.4D we simulated a pre-synaptic burst consisting of 5 spikes with 20 ms time difference, and a post-synaptic burst of 4 spikes, also with 20 ms time difference. The starting points of these bursts were shifted relative to each other. We kept the weight fixed at $w = 3.5$ for $c = e^{-5}$, and the learning rate fixed at $\eta = 0.1$, and added up the resulting weight changes for all 4 postsynaptic spikes.

Simulation for Fig. 4.6:

$\mu_{ou} = 250, A_{inh} = 2000, O_{inh} = 400.$

Simulation for Fig. 4.7 and Suppl. Fig. 4.11:

$\mu_{ou} = 250, A_{inh} = 1500, O_{inh} = 1000.$

Simulation for Suppl. Fig. 4.10:

$\mu_{ou} = 250, A_{inh} = 2700, O_{inh} = 400.$

Acknowledgments

We would like to thank Wulfram Gerstner for critical comments to an earlier version of this paper. MP would like to thank Rodney J. Douglas and Tobi Delbruck for their generous advice and support.

5

Neural Dynamics as Sampling

The organization of computations in networks of spiking neurons in the brain is still largely unknown, in particular in view of the inherently stochastic features of their firing activity and the experimentally observed trial-to-trial variability of neural systems in the brain. In principle there exists a powerful computational framework for stochastic computations, probabilistic inference by sampling, which can explain a large number of macroscopic experimental data in neuroscience and cognitive science. But it has turned out to be surprisingly difficult to create a link between these abstract models for stochastic computations and more detailed models of the dynamics of networks of spiking neurons. Here we create such a link, and show that under some conditions the stochastic firing activity of networks of spiking neurons can be interpreted as probabilistic inference via Markov chain Monte Carlo (MCMC) sampling. Since common methods for MCMC sampling in distributed systems, such as Gibbs sampling, are inconsistent with the dynamics of spiking neurons, we introduce a different approach based on non-reversible Markov chains, that is able to reflect inherent temporal processes of spiking neuronal activity through a suitable choice of random variables. We propose a neural network model and show by a rigorous theoretical analysis that its neural activity implements MCMC sampling of a given distribution, both for the case of discrete and continuous time. This provides a step towards closing the gap between abstract functional models of cortical computation and more detailed models of networks of spiking

neurons.

5.1 Author Summary

It is well-known that neurons communicate with short electric pulses, called action potentials or spikes. But how can spiking networks implement complex computations? Attempts to relate spiking network activity to results of deterministic computation steps, like the output bits of a processor in a digital computer, are conflicting with findings from cognitive science and neuroscience, the latter indicating the neural spike output in identical experiments changes from trial to trial, i.e., neurons are "unreliable". Therefore, it has been recently proposed that neural activity should rather be regarded as *samples* from an underlying *probability distribution* over many variables which, e.g., represent a model of the external world incorporating prior knowledge, memories as well as sensory input. This hypothesis assumes that networks of stochastically spiking neurons are able to emulate powerful algorithms for reasoning in the face of uncertainty, i.e., to carry out probabilistic inference. In this work we propose a detailed neural network model that indeed fulfills these computational requirements and we relate the spiking dynamics of the network to concrete probabilistic computations. Our model suggests that neural systems are suitable to carry out probabilistic inference by using stochastic, rather than deterministic, computing elements.

5.2 Introduction

Attempts to understand the organization of computations in the brain from the perspective of traditional, mostly deterministic, models of computation, such as attractor neural networks or Turing machines, have run into problems: Experimental data suggests that neurons, synapses, and neural systems are inherently *stochastic* [189], especially in vivo, and therefore seem less suitable for implementing deterministic computations. This holds for ion channels of neurons [28], synaptic release [64], neural response to stimuli (trial-to-trial variability) [13, 73], and perception [24]. In fact, several experimental studies arrive at the conclusion that external stimuli only modulate the highly stochastic spontaneous firing activity of cortical networks of neurons [63, 188]. Furthermore, traditional models for neural computation have been challenged by the fact that

typical sensory data from the environment is often noisy and ambiguous, hence requiring neural systems to take *uncertainty* about external inputs into account. Therefore many researchers have suggested that information processing in the brain carries out probabilistic, rather than logical, inference for making decisions and choosing actions [54, 70, 72, 81, 82, 86, 104, 121, 127, 136, 183, 193, 215, 232]. Probabilistic inference has emerged in the 1960's [171], as a principled mathematical framework for reasoning in the face of uncertainty with regard to observations, knowledge, and causal relationships, which is characteristic for real-world inference tasks. This framework has become tremendously successful in real-world applications of artificial intelligence and machine learning. A typical computation that needs to be carried out for probabilistic inference on a high-dimensional joint distribution $p(z_1, \ldots, z_l, z_{l+1}, \ldots, z_K)$ is the evaluation of the conditional distribution $p(z_1, \ldots, z_l | z_{l+1}, \ldots, z_K)$ (or marginals thereof) over some variables of interest, say $z_1, \ldots, z_l$, given variables $z_{l+1}, \ldots, z_K$. In the following, we will call the set of variables $z_{l+1}, \ldots, z_K$, which we condition on, the *observed* variables and denote it by $\mathbf{o}$.

Numerous studies in different areas of neuroscience and cognitive science have suggested that probabilistic inference could explain a variety of computational processes taking place in neural systems (see [54, 183]). In models of perception the observed variables $\mathbf{o}$ are interpreted as the sensory input to the central nervous system (or its early representation by the firing response of neurons, e.g., in the LGN in the case of vision), and the variables $z_1, \ldots, z_l$ model the interpretation of the sensory input, e.g., the texture and position of objects in the case of vision, which might be encoded in the response of neurons in various higher cortical areas [136]. Furthermore, in models for motor control the observed variables $\mathbf{o}$ often consist not only of sensory and proprioceptive inputs to the brain, but also of specific goals and constraints for a planned movement [67, 221, 222], whereas inference is carried out over the variables $z_1, \ldots, z_l$ representing a motor plan or motor commands to muscles. Recent publications show that human reasoning and learning can also be cast into the form of probabilistic inference problems [87, 164, 218]. In these models learning of concepts, ranging from concrete to more abstract ones, is interpreted as inference in lower and successively higher levels of hierarchical probabilistic models, giving a consistent description of inductive learning within and across domains of knowledge.

In spite of this active research on the functional level of neural processing, it turned out to be surprisingly hard to relate the computational machinery required for probabilistic inference to experimental data on neurons, synapses, and neural systems. There are mainly two different approaches for implementing the computational machinery for probabilistic inference in "neural hardware". The first class of approaches builds on deterministic methods for evaluating exactly or approximately the desired conditional and/or marginal distributions, whereas the second class relies on sampling from the probability distributions in question. Multiple models in the class of deterministic approaches implement algorithms from machine learning called message passing or belief propagation [45, 141, 182, 211]. By clever reordering of sum and product operators occurring in the evaluation of the desired probabilities, the total number of computation steps are drastically reduced. The results of subcomputations are propagated as "messages" or "beliefs" that are sent to other parts of the computational network. Other deterministic approaches for representing distributions and performing inference are probabilistic population code (PPC) models [194]. Although deterministic approaches provide a theoretically sound hypothesis about how complex computations can possibly be embedded in neural networks and explain aspects of experimental data, it seems difficult (though not impossible) to conciliate them with other aspects of experimental evidence, such as stochasticity of spiking neurons, spontaneous firing, trial-to-trial variability, and perceptual multistability.

Therefore other researchers (e.g., [62, 72, 104, 215]) have proposed to model computations in neural systems as probabilistic inference based on a different class of algorithms, which requires stochastic, rather than deterministic, computational units. This approach, commonly referred to as sampling, focuses on drawing *samples*, i.e., concrete values for the random variables that are distributed according to the desired probability distribution. Sampling can naturally capture the effect of apparent stochasticity in neural responses and seems to be furthermore consistent with multiple experimental effects reported in cognitive science literature [72, 215]. On the conceptual side, it has proved to be difficult to implement learning in message passing and PPC network models. In contrast, following the lines of [3], the sampling approach might be well suited to incorporate learning.

Previous network models that implement sampling in neural networks are mostly based on a special sampling algorithm called Gibbs (or general Metropolis-Hastings)

sampling [70, 72, 100, 215]. The dynamics that arise from this approach, the so-called Glauber dynamics, however are only superficially similar to spiking neural dynamics observed in experiments, rendering these models rather abstract. Building on and extending previous models, we propose here a family of network models, that can be shown to exactly sample from any arbitrary member of a well-defined class of probability distributions via their inherent network dynamics. These dynamics incorporate refractory effects and finite durations of postsynaptic potentials (PSPs), and are therefore more biologically realistic than existing approaches. Formally speaking, our model implements Markov chain Monte Carlo (MCMC) sampling in a spiking neural network. In contrast to prior approaches however, our model incorporates irreversible dynamics (i.e., no detailed balance) allowing for finite time PSPs and refractory mechanisms. Furthermore, we also present a continuous time version of our network model. The resulting stochastic dynamical system can be shown to sample from the correct distribution. In general, continuous time models arguably provide a higher amount of biological realism compared to discrete time models.

The paper is structured in the following way. First we provide a brief introduction to MCMC sampling. We then define the neural network model whose neural activity samples from a given class of probability distributions. The model will be first presented in discrete time together with some illustrative simulations. An extension of the model to networks of more detailed spiking neuron models which feature a relative refractory mechanism is presented. Furthermore, it is shown how the neural network model can also be formulated in continuous time. Finally, as a concrete simulation example we present a simple network model for perceptual multistability.

5.3 Results

5.3.1 Recapitulation of MCMC sampling

In machine learning, sampling is often considered the "gold standard" of inference methods, since, assuming that we can sample from the distribution in question, and assuming enough computational resources, any inference task can be carried out with arbitrary precision (in contrast to some deterministic approximate inference methods such as variational inference). However sampling from an arbitrary distribution can be a difficult problem in itself, as, e.g., many distributions can only be evaluated modulo

a global constant (the partition function). In order to circumvent these problems, elaborate MCMC sampling techniques have been developed in machine learning and statistics [5]. MCMC algorithms are based on the following idea: instead of producing an ad-hoc sample, a process that is heuristically comparable to a global search over the whole state space of the random variables, MCMC methods produce a new sample via a "local search" around a point in the state space that is already (approximately) a sample from the distribution.

More formally, a Markov chain M (in discrete time) is defined by a set S of states (we consider for discrete time only the case where S has a finite size, denoted by $|S|$) together with a transition operator T. The operator T is a conditional probability distribution $T(s|s')$ over the next state s given a preceding state s'. The Markov chain M is started in some initial state $s(0)$, and moves through a trajectory of states $s(t)$ via iterated application of the stochastic transition operator T. More precisely, if $s(t-1)$ is the state at time $t-1$, then the next state $s(t)$ is drawn from the conditional probability distribution $T(s|s(t-1))$. An important theorem from probability theory (see, e.g., p. 232 in [89]) states that if M is irreducible (i.e., any state in S can be reached from any other state in S in finitely many steps with probability > 0) and aperiodic (i.e., its state transitions cannot be trapped in deterministic cycles), then the probability $p(s(t) = s|s(0))$ converges for $t \to \infty$ to a probability $p(s)$ that does not depend on the initial state $s(0)$. This state distribution p is called the invariant distribution of M. The irreducibility of M implies that it is the only distribution over the states S that is invariant under its transition operator T, i.e.

$$p(s) = \sum_{s' \in S} T(s|s') \cdot p(s') \quad . \tag{5.1}$$

Thus, in order to carry out probabilistic inference for a given distribution p, it suffices to construct an irreducible and aperiodic Markov chain M that leaves p invariant, i.e., satisfies equation (5.1). Then one can answer numerous probabilistic inference questions regarding p without any numerical computations of probabilities. Rather, one plugs in the observed values for some of the random variables (RVs) and simply collects samples from the conditional distribution over the other RVs of interest when the Markov chain approaches its invariant distribution.

A convenient and popular method for the construction of an operator T for a given distribution p is looking for operators T that satisfy the following detailed balance

condition,

$$T(s|s') \cdot p(s') = T(s'|s) \cdot p(s) \tag{5.2}$$

for all s, $s' \in S$. A Markov chain that satisfies (5.2) is said to be reversible. In particular, the Gibbs and Metropolis-Hastings algorithms employ reversible Markov chains. A very useful property of (5.2) is that it implies the invariance property (5.1), and this is in fact the standard method for proving (5.1). However, as our approach makes use of irreversible Markov chains as explained below, we will have to prove (5.1) directly.

5.3.2 Neural sampling

Let $p(z_1, \ldots, z_K)$ be some arbitrary joint distribution over K binary variables $z_1, \ldots, z_K$ that only takes on values > 0. We will show that under a certain computability assumption on p a network $\mathcal{N}$ consisting of K spiking neurons $\nu_1, \ldots, \nu_K$ can sample from p using its inherent stochastic dynamics. More precisely, we show that the stochastic firing activity of $\mathcal{N}$ can be viewed as a non-reversible Markov chain that samples from the given probability distribution p. If a subset $\mathbf{o}$ of the variables are observed, modelled as the corresponding neurons being "clamped" to the observed values, the remaining network samples from the conditional distribution of the remaining variables given the observables. Hence, this approach offers a quite natural implementation of probabilistic inference. It is similar to sampling approaches which have already been applied extensively, e.g., in Boltzmann machines, however our model is more biologically realistic as it incorporates aspects of the inherent temporal dynamics and spike-based communication of a network of spiking neurons. We call this approach *neural sampling* in the remainder of the paper.

In order to enable a network $\mathcal{N}$ of spiking neurons to sample from a distribution $p(z_1, \ldots, z_K)$ of binary variables z_k, one needs to specify how an assignment $(z_1, \ldots, z_K) \in \{0, 1\}^K$ of values to these binary variables can be represented by the spiking activity of the network $\mathcal{N}$ and vice versa. A spike, or action potential, of a biological neuron ν_k has a short duration of roughly 1 ms. But the effect of such spike, both on the neuron ν_k itself (in the form of refractory processes) and on the membrane potential of other neurons (in the form of postsynaptic potentials) lasts substantially longer, on the order of 5 ms to 100 ms. In order to capture this temporally extended effect of

each spike, we fix some parameter τ that models the average duration of these temporally extended processes caused by a spike. We say that a binary vector $(z_1, \ldots, z_K)$ is represented by the firing activity of the network $\mathcal{N}$ at time t for $k = 1, \ldots, K$ iff:

$$z_k(t) = 1 \quad \Leftrightarrow \quad \nu_k \text{ has fired within the time interval } (t - \tau,\, t]. \tag{5.3}$$

In other words, any spike of neuron ν_k sets the value of the associated binary variable z_k to 1 for a duration of length τ.

An obvious consequence of this definition is that the binary vector $(z_1, \ldots, z_K)$ that is defined by the activity of $\mathcal{N}$ at time t does not fully capture the internal state of this stochastic system. Rather, one needs to take into account additional non-binary variables $(\zeta_1, \ldots, \zeta_K)$, where the value of ζ_k at time t specifies *when* within the time interval $(t-\tau,\, t]$ the neuron ν_k has fired (if it has fired within this time interval, thereby causing $z_k = 1$ at time t). The neural sampling process has the Markov property only with regard to these more informative auxiliary variables $\zeta_1, \ldots, \zeta_K$. Therefore our analysis of neural sampling will focus on the temporal evolution of these auxiliary variables. We adopt the convention that each spike of neuron ν_k sets the value of ζ_k to its maximal value τ, from which it linearly decays back to 0 during the subsequent time interval of length τ.

For the construction of the sampling network $\mathcal{N}$, we assume that the membrane potential $u_k(t)$ of neuron ν_k at time t equals the log-odds of the corresponding variable z_k to be active, and refer to this property as *neural computability condition*:

$$u_k(t) = \log \frac{p(z_k = 1 | \mathbf{z}_{\setminus k})}{p(z_k = 0 | \mathbf{z}_{\setminus k})} \quad , \tag{5.4}$$

where we write z_k for $z_k(t)$ and $\mathbf{z}_{\setminus k}$ for the current values $z_i(t)$ of all other variables z_i with $i \neq k$. Under the assumption we make in equation (5.4), i.e., that the neural membrane potential reflects the log-odds of the corresponding variable z_k, it is required that each single neuron in the network can actually compute the right-hand side of equation (5.4), i.e., that it fulfills the neural computability condition.

A concrete class of probability distributions, that we will use as an example in the remainder, are Boltzmann distributions:

$$p(\mathbf{z}) = \frac{1}{Z} \, \exp\left(\sum_{i,j} \frac{1}{2} W_{ij} z_i z_j + \sum_i b_i z_i \right) \tag{5.5}$$

with arbitrary real valued parameters b_i, W_{ij} which satisfy $W_{ij} = W_{ji}$ and $W_{ii} = 0$ (the constant Z ensures the normalization of $p(\mathbf{z})$). For the Boltzmann distribution, condition (5.4) is satisfied by neurons ν_k with the standard membrane potential

$$u_k(t) = b_k + \sum_{i=1}^{K} W_{ki} z_i(t) \quad , \tag{5.6}$$

where b_k is the bias of neuron ν_k (which regulates its excitability), W_{ki} is the strength of the synaptic connection from neuron ν_i to ν_k, and $W_{ki} z_i(t)$ approximates the time course of the postsynaptic potential in neuron ν_k caused by a firing of neuron ν_i with a constant signal of duration τ (i.e., a square pulse). As we will describe below, spikes of neuron ν_k are evoked stochastically depending on the current membrane potential u_k and the auxiliary variable ζ_k.

The neural computability condition (5.4) links classes of probability distributions to neuron and synapse models in a network of spiking neurons. As shown above, Boltzmann distributions satisfy the condition if one considers point neuron models which compute a linear weighted sum of the presynaptic inputs. The class of distributions can be extended to include more complex distributions using a method proposed in [158] which is based on the following idea. Neuron ν_k representing the variable z_k is not directly influenced by the activities $\mathbf{z}_{\setminus k}$ of the presynaptic neurons, but via intermediate nonlinear preprocessing elements. This preprocessing might be implemented by dendrites or other (inter-) neurons and is assumed to compute nonlinear combinations of the presynaptic activities $\mathbf{z}_{\setminus k}$ (similar to a kernel). This allows the membrane potential u_k, and therefore the log-odds ratio on the right-hand side of (5.4), to represent a more complex function of the activities $\mathbf{z}_{\setminus k}$, giving rise to more complex joint distributions $p(\mathbf{z})$. The concrete implementation of non-trivial directed and undirected graphical models with the help of preprocessing elements in the neural sampling framework is subject of current research. For the examples given in this study, we focus on the standard form of the membrane potential (5.6) of point neurons. As shown below, these spiking network models can emulate any Boltzmann machine (BM) [3].

A substantial amount of preceding studies has demonstrated that BMs are very powerful, and that the application of suitable learning algorithms for setting the weights W_{ij} makes it possible to learn and represent complex sensory processing tasks by such distributions [98, 100]. In applications in statistics and machine learning using such

Boltzmann distributions, sampling is typically implemented by Gibbs sampling or more general *reversible* MCMC methods. However, it is difficult to model some neural processes, such as an absolute refractory period or a postsynaptic potential (PSP) of fixed duration, using a reversible Markov chain, but they are more conveniently modelled using an irreversible one. As we wish to keep the computational power of BMs and at the same time to augment the sampling procedure with aspects of neural dynamics (such as PSPs with fixed durations, refractory mechanisms) to increase biological realism, we focus in the following on irreversible MCMC methods (keeping in mind that this might not be the only possible way to achieve these goals).

5.3.3 Neural sampling in discrete time

Here we describe neural dynamics in discrete time with an absolute refractory period τ. We interpret one step of the Markov chain as a time step dt in biological real time. The dynamics of the variable ζ_k, that describes the time course of the effect of a spike of neuron ν_k, are defined in the following way. ζ_k is set to the value τ when neuron ν_k fires, and decays by 1 at each subsequent discrete time step. The parameter τ is chosen to be some integer, so that ζ_k decays back to 0 in exactly τ time steps. The neuron can only spike (with a probability that is a function of its current membrane potential u_k) if its variable $\zeta_k \leq 1$. If however, $\zeta_k > 1$, the neuron is considered refractory and it cannot spike, but its ζ_k is reduced by 1 per time step. To show that these simple dynamics do indeed sample from the given distribution $p(\mathbf{z})$, we proceed in the following way. We define a joint distribution $p(\zeta, \mathbf{z})$ which has the desired marginal distribution $\sum_\zeta p(\zeta, \mathbf{z}) = p(\mathbf{z})$. Further we formalize the dynamics informally described above as a transition operator T operating on the state vector $(\zeta, \mathbf{z})$. Finally, in the Methods section, we show that $p(\zeta, \mathbf{z})$ is the unique invariant distribution of this operator T, i.e., that the dynamics described by T produce samples $\mathbf{z}$ from the desired distribution $p(\mathbf{z})$. We refer to sampling through networks with this stochastic spiking mechanism as *neural sampling with absolute refractory period* due to the persistent refractory process.

Given the distribution $p(\mathbf{z})$ that we want to sample from, we define the following

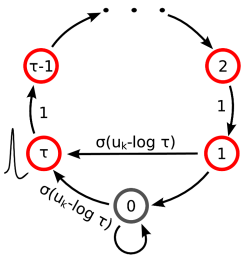

Figure 5.1: Neuron model with absolute refractory mechanism. The figure shows a schematic of the transition operator T^k for the internal state variable ζ_k of a spiking neuron ν_k with an absolute refractory period. The neuron can fire in the resting state $\zeta_k = 0$ and in the last refractory state $\zeta_k = 1$.

joint distribution $p(\zeta, \mathbf{z})$ over the neural variables:

$$\begin{aligned} p(\zeta, \mathbf{z}) &:= p(\zeta|\mathbf{z}) \cdot p(\mathbf{z}) \quad \text{with} \quad p(\zeta|\mathbf{z}) := \prod_{k=1}^{K} p(\zeta_k|z_k) \\ \text{where} \quad p(\zeta_k|z_k) &:= \begin{cases} \tau^{-1} & \text{for} \quad z_k = 1 \ \wedge \ \zeta_k > 0 \\ 1 & \text{for} \quad z_k = 0 \ \wedge \ \zeta_k = 0 \\ 0 & \text{otherwise} \quad . \end{cases} \end{aligned} \tag{5.7}$$

This definition of $p(\zeta_k|z_k)$ simply expresses that if $z_k = 1$, then the auxiliary variable ζ_k can assume any value in $\{1, 2, \ldots, \tau\}$ with equal probability. On the other hand ζ_k necessarily assumes the value 0 if $z_k = 0$ (i.e., when the neuron is in its resting state).

The state transition operator T can be defined in a transparent manner as a composition of K transition operators, $T = T^1 \circ \ldots \circ T^K$, where T^k only updates the variables ζ_k and z_k of neuron ν_k, i.e., the neurons are updated sequentially in the same order (this severe restriction will become obsolete in the case of continuous time discussed below). We define the composition as $(T^k \circ T^l)(\cdot) = (T^k(T^l(\cdot))$, i.e., T^l is applied prior to T^k. The new values of ζ_k and z_k only depend on the previous value ζ_k' and on the current membrane potential $u_k(\mathbf{z}_{\setminus k})$. The interesting dynamics take place in the variable ζ_k. They are illustrated in Figure 5.1, where the arrows represent transition probabilities greater than 0.

If the neuron ν_k is not refractory, i.e., $\zeta_k' \leq 1$, it can spike (i.e., a transition from $\zeta_k' \leq 1$ to $\zeta_k = \tau$) with probability

$$T^k(\zeta_k = \tau|\zeta_k', \mathbf{z}_{\setminus k}) = \sigma(u_k - \log \tau) \quad , \tag{5.8}$$

where $\sigma(x) = (1 + e^{-x})^{-1}$ is the standard sigmoidal activation function and the log denotes the natural logarithm. The term u_k is the current membrane potential, which depends on the current values of the variables z_i for $i \neq k$. The term $\log \tau$ in (5.8) reflects the granularity of a chosen discrete time scale. If it is very fine (say one step equals one microsecond), then τ is large, and the firing probability at each specific discrete time step is therefore reduced. If the neuron in a state with $\zeta'_k \leq 1$ does not spike, ζ_k relaxes into the resting state $\zeta_k = 0$ corresponding to a non-refractory neuron.

If the neuron is in a refractory state, i.e., $\zeta'_k > 1$, its new variable ζ_k assumes deterministically the next lower value $\zeta_k = \zeta'_k - 1$, reflecting the inherent temporal process:

$$T^k(\zeta_k = \zeta'_k - 1 | \zeta'_k, \mathbf{z}_{\setminus k}) = 1 \quad . \tag{5.9}$$

After the transition of the auxiliary variable ζ_k, the binary variable z_k is deterministically set to a consistent state, i.e., $z_k = 1$ if $\zeta_k \geq 1$ and $z_k = 0$ if $\zeta_k = 0$.

It can be shown that each of these stochastic state transition operators T^k leaves the given distribution p invariant, i.e., satisfies equation (5.1). This implies that any composition or mixture of these operators T^k also leaves p invariant, see, e.g., [5]. In particular, the composition $T = T^1 \circ \ldots \circ T^K$ of these operators T^k leaves p invariant, which has a quite natural interpretation as firing dynamics of the spiking neural network $\mathcal{N}$: At each discrete time step the variables ζ_k, z_k are updated for all neurons ν_k, where the update of ζ_k, z_k takes preceding updates for ζ_i, z_i with $i > k$ into account. Alternatively, one could also choose at each discrete time step a different order for updates according to [5]. The assumption of a well-regulated updating policy will be overcome in the continuous-time limit, i.e., in case where the neural dynamics are described as a Markov jump process. In the methods section we prove the following central theorem:

Theorem 1. *$p(\zeta, \mathbf{z})$ is the unique invariant distribution of operator T, i.e., T is aperiodic and irreducible and satisfies*

$$p(\zeta, \mathbf{z}) = \sum_{\zeta', \mathbf{z}'} T(\zeta, \mathbf{z} | \zeta', \mathbf{z}') \cdot p(\zeta', \mathbf{z}') \quad . \tag{5.10}$$

The proof of this Theorem is provided by Lemmata 1 – 3 in the Methods section. The statement that T (which is composed of the operators T^k) is irreducible and

aperiodic ensures that p is the *unique* invariant distribution of the Markov chain defined by T, i.e., that irrespective of the initial network state the successive application of T explores the whole state space in a non-periodic manner.

This theorem guarantees that after a sufficient "burn-in" time (more precisely in the limit of an infinite "burn-in" time), the dynamics of the network, which are given by the transition operator T, produce samples from the distribution $p(\zeta, \mathbf{z})$. As by construction $\sum_\zeta p(\zeta, \mathbf{z}) = p(\mathbf{z})$, the Markov chain provides samples from the given distribution $p(\mathbf{z})$. Furthermore, the network $\mathcal{N}$ can carry out probabilistic inference for this distribution. For example, $\mathcal{N}$ can be used to sample from the posterior distribution $p(z_1 \ldots, z_l | z_{l+1}, \ldots, z_K)$ over $z_1 \ldots, z_l$ given $z_{l+1}, \ldots, z_K$. One just needs to clamp those neurons $\nu_{l+1}, \ldots, \nu_K$ to the corresponding observed values. This could be implemented by injecting a strong positive (negative) current into the units with $z_j = 1$ ($z_j = 0$). Then, as soon as the stochastic dynamics of $\mathcal{N}$ has converged to its invariant distribution, the averaged firing rate of neuron ν_1 is proportional to the following desired marginal probability

$$p(z_1 = 1 | z_{l+1}, \ldots, z_K) = \sum_{z_2, \ldots, z_l} p(z_1 = 1, z_2, \ldots, z_l | z_{l+1}, \ldots, z_K) \quad .$$

In a biological neural system this result of probabilistic inference could for example be read out by an integrator neuron that counts spikes from this neuron ν_1 within a behaviorally relevant time window of a few hundred milliseconds, similarly as the experimentally reported integrator neurons in area LIP of monkey cortex [81, 232]. Another readout neuron that receives spike input from ν_k could at the same time estimate $p(z_k = 1 | z_{l+1}, \ldots, z_K)$ for another RV z_k. But valuable information for probabilistic inference is not only provided by firing rates or spike counts, but also by spike correlations of the neurons $\nu_1, \ldots, \nu_l$ in $\mathcal{N}$. For example, the probability $p(z_1 = 1, z_2 = 1 | z_{l+1}, \ldots, z_K)$ can be estimated by a readout neuron that responds to superpositions of EPSPs caused by near-coincident firing of neurons ν_1 and ν_2 within a time interval of length τ. Thus, a large number of different probabilistic inferences can be carried out efficiently in parallel by readout neurons that receive spike input from different subsets of neurons in the network $\mathcal{N}$.

5.3.3.1 Variation of the discrete time model with a relative refractory mechanism

For the previously described simple neuron model, the refractory process was assumed to last for τ time steps, exactly as long as the postsynaptic potentials caused by each spike. In this section we relax this assumption by introducing a more complex and biologically more realistic neuron model, where the duration of the refractory process is decoupled from the duration τ of a postsynaptic potential. Thus, this model can for example also fire bursts of spikes with an interspike interval $< \tau$. The introduction of this more complex neuron model comes at the price that one can no longer prove that a network of such neurons samples from the desired distribution p. Nevertheless, if the sigmoidal activation function σ is replaced by a different activation function f, one can still prove that the sampling is "locally correct", as specified in equation (5.12) below. Furthermore, our computer simulations suggest that also globally the error introduced by the more complex neuron model is not functionally significant, i.e that statistical dependencies between the RVs $\mathbf{z}$ are still faithfully captured.

The neuron model with a relative refractory period is defined in the following way. Consider some arbitrary refractory function $g : [0, \ldots, \tau] \to \mathbb{R}$ with $g(\tau) = 0,\ g(0) = 1$, and $g(l) \geq 0$ for $l = 1, \ldots, \tau - 1$. The idea is that $g(\zeta_k)$ models the readiness of the neuron to fire in its state ζ_k. This readiness has value 0 when the neuron has fired at the preceding time step (i.e., $\zeta_k = \tau$), and assumes the resting state 1 when ζ_k has dropped to 0. In between, the readiness may take on any non-negative value according to the function $g(\zeta_k)$. The function g does not need to be monotonic, allowing for example that it increases to high values in between, yielding a preferred interspike interval of a oscillatory neuron. The firing probability of neuron ν_k in state ζ_k is given by $g(\zeta_k) \cdot f(u_k)$, where $f(u_k)$ is an appropriate function of the membrane potential as described below. Thus this function g is closely related to the function η (called afterpotential) in the spike response model [73] as well as to the self-excitation kernel in Generalized Linear Models [176]. In general, different neurons in the network may have different refractory profiles, which can be modeled by a different refractory function for each neuron ν_k. However for the sake of notational simplicity we assume a single refractory function in the following.

In the presence of this refractory function g one needs to replace the sigmoidal activation function $\sigma(u_k - \log \tau)$ by a suitable function $f(u_k)$ that satisfies the condition

$$\exp(u) = f(u)\frac{\sum_{\eta=1}^{\tau}\prod_{\zeta=\eta+1}^{\tau}(1-g(\zeta)\cdot f(u))}{\prod_{\zeta=1}^{\tau}(1-g(\zeta)\cdot f(u))} \tag{5.11}$$

for all real numbers u. This equation can be derived (see Methods section Lemma 5) if one requires each neuron ν_k to represent the correct distribution $p(z_k|\mathbf{z}_{\setminus k})$ over z_k conditioned the variables $\mathbf{z}_{\setminus k}$. One can show that, for any g as above, there always exists a continuous, monotonic function f which satisfies this equation (see Lemma 4 in Methods). Unfortunately (5.11) cannot be solved analytically for f in general. Hence, for simulations we approximate the function f for a given g by numerically solving (5.11) on a grid and interpolating between the grid points with a constant function. Examples for several functions g and the associated f are shown in Figure 5.2B and Figure 5.2C respectively. Furthermore, spike trains emitted by single neurons with these refractory functions g and the corresponding functions f are shown in Figure 5.2D for the case of piecewise constant membrane potentials. This figure indicates, that functions g that define a shorter refractory effect lead to higher firing rates and more irregular firing. It is worth noticing that the standard activation function $\sigma(u_k - \log\tau)$ is the solution of equation (5.11) for the absolute refractory function, i.e., for $g(0) = g(1) = 1$ and $g(l) = 0$ for $1 < l \leq \tau$.

The transition operator T^k is defined for this model in a very similar way as before. However, for $1 < \zeta_k' \leq \tau$, when the variable ζ_k' was deterministically reduced by 1 in the simpler model (yielding $\zeta_k = \zeta_k' - 1$), this reduction occurs now only with probability $1 - g(\zeta_k') \cdot f(u_k)$. With probability $g(\zeta_k') \cdot f(u_k)$ the operator T^k sets $\zeta_k = \tau$, modeling the firing of another spike of neuron ν_k at this time point. The neural computability condition (5.4) remains unchanged, e.g., $u_k = b_k + \sum_{i=1}^{K} W_{ki} z_i$ for a Boltzmann distribution. A schema of the stochastic dynamics of this local state transition operator $T^k(\zeta_k|\zeta_k', \mathbf{z}'_{\setminus k})$ is shown in Figure 5.2A.

This transition operator T^k has the following properties. In Lemma 5 in Methods it is proven that the unique invariant distribution of T^k, denoted as $q_k^*(\zeta_k, z_k|\zeta_{\setminus k}, \mathbf{z}_{\setminus k})$, gives rise to the correct marginal distribution over z_k, i.e.

$$\sum_{\zeta_k=0}^{\tau} q_k^*(\zeta_k, z_k|\zeta_{\setminus k}, \mathbf{z}_{\setminus k}) = p(z_k|\mathbf{z}_{\setminus k}) \quad .$$

This means that a neuron whose dynamics is described by T^k samples from the correct distribution $p(z_k|\mathbf{z}_{\setminus k})$ if it receives a static input from the other neurons in the network,

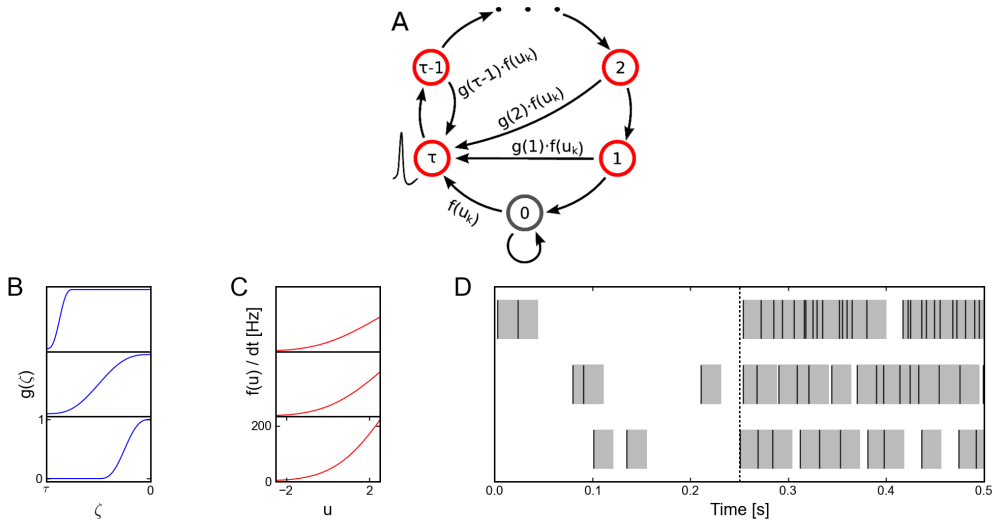

Figure 5.2: Neuron model with relative refractory mechanism. The figure shows the transition operator T^k, refractory functions g and activation functions f for the neuron model with relative refractory mechanism. (A) Transition probabilities of the internal variable ζ_k given by T^k. (B) Three examples of possible refractory functions g. They assume value 0 when the neuron cannot spike, and return to value 1 (full readiness to fire again) with different time courses. The value of g at intermediate time points regulates the current probability of firing of neuron ν_k (see A). The x-axis is equivalent to the number of time steps since last spike (running from 0 to τ from left to right). (C) Associated activation functions f according to (5.11). (D) Spike trains produced by the resulting three different neuron models with (hypothetical) membrane potentials that jump at time 0.25 s from a constant low value to a constant high value. Black horizontal bars indicate spikes, and the active states $z_k = 1$ are indicated by gray shaded areas of duration $\tau \cdot dt = 20$ms after each spike. It can be seen from this example that different refractory mechanisms give rise to different spiking dynamics.

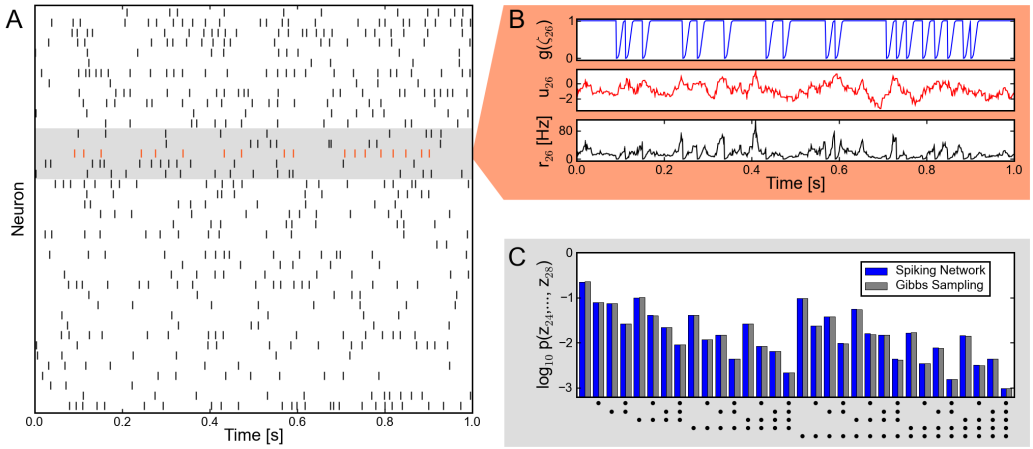

Figure 5.3: Sampling from a Boltzmann distribution by spiking neurons with relative refractory mechanism. (A) Spike raster of the network. (B) Traces of internal state variables of a neuron (# 26, indicated by orange spikes in A). The rich interaction of the network gives rise to rapidly changing membrane potentials and instantaneous firing rates. (C) Joint distribution of 5 neurons (gray shaded area in A) obtained by the spiking neural network and Gibbs sampling from the same distribution. Active states $z_i = 1$ are indicated by a black dot, using one row for each neuron ν_i, the columns list all $2^5 = 32$ possible states $(z_{24}, \ldots, z_{28})$ of these 5 neurons. The tight match between both distributions suggests that the spiking network represents the target probability distribution p with high accuracy.

i.e., as long as its membrane potential u_k is constant. Hence the "local" computation performed by such neuron can be considered as correct. If however, several neurons in the network change their states in a short interval of time, the joint distribution over $\mathbf{z}$ is in general not the desired one, i.e., $\sum_\zeta q^*(\zeta, \mathbf{z}) \neq p(\mathbf{z})$, where $q^*(\zeta, \mathbf{z})$ denotes the invariant distribution of $T = T^1 \circ \ldots \circ T^K$. In the Methods section, we present simulation results that indicate that the error of the approximation to the desired Boltzmann distributions introduced by neural sampling with relative refractory mechanism is rather minute. It is shown that the neural sampling approximation error is orders of magnitudes below the one introduced by a fully factorized distribution (which amounts to assuming correct marginal distributions $p(z_k)$ and independent neurons).

To illustrate the sampling process with the relative refractory mechanism, we examine a network of $K = 40$ neurons. We aim to sample from a Boltzmann distribution (5.5) with parameters W_{ij}, b_i being randomly drawn from normal distributions. For the neuron model, we use the relative refractory mechanism shown in the mid row of

Figure 5.2B. A detailed description of the simulation and the parameters used is given in the Methods section. A spike pattern of the resulting sampling network is shown in Figure 5.3A. The network features a sparse, irregular spike response with average firing rate of 13.9 Hz. For one neuron ν_{26}, indicated with orange spikes, the internal dynamics are shown in Figure 5.3B. After each action potential the neuron's refractory function $g(\zeta_{26})$ drops to zero and reduces the probability of spiking again in a short time interval. The influence of the remaining network $\mathbf{z}_{\setminus 26}$ is transmitted to neuron ν_{26} via PSPs of duration $\tau \cdot dt = 20\,\text{ms}$ and sums up to the fluctuating membrane potential u_{26}. As reflected in the highly variable membrane potential even this small network exhibits rich interactions. To represent the correct distribution $p(z_{26}|\mathbf{z}_{\setminus 26})$ over z_{26} conditioned on $\mathbf{z}_{\setminus 26}$, the neuron ν_{26} continuously adapts its instantaneous firing rate. To quantify the precision with which the spiking network draws samples from the target distribution (5.5), Figure 5.3C shows the joint distribution of 5 neurons. For comparison we accompany the distribution of sampled network states with the result obtained from the standard Gibbs sampling algorithm (considered as the ground truth). Since the number of possible states $\mathbf{z}$ grows exponentially in the number of neurons, we restrict ourselves for visualization purposes to the distribution $p(z_{24}, \ldots, z_{28})$ of the gray shaded units and marginalize over the remaining network. The probabilities are estimated from 10^7 samples, i.e., from 10^7 successive states $\mathbf{z}$ of the Markov chain. Stochastic deviations of the estimated probabilities due to the finite number of samples are quite small (typical errors $\Delta p(\mathbf{z})/\sqrt{p(\mathbf{z})} \approx 10^{-3}$) and are comparable to systematic deviations due to the only locally correct computation of neurons with relative refractory mechanism. In the Methods section, we present further simulation results showing that the proposed networks consisting of neurons with relative refractory mechanism approximate the desired target distributions faithfully over a large range of distribution parameters.

In order to illustrate that the proposed sampling networks feature biologically quite realistic spiking dynamics, we present in the Methods section several neural firing statistics (e.g., the inter-spike interval histogram) of the network model. In general, the statistics computed from the model match experimentally observed statistics well. The proposed network models are based on the assumption of rectangular-shaped, renewal PSPs. More precisely, we define renewal (or non-additive) PSPs in the following way. Renewal PSPs evoked by a single synapse do not add up but are merely prolonged in their duration (according to equation (5.6)); renewal PSPs elicited at different synapses

nevertheless add up in the normal way. In Methods we investigate the impact of replacing the theoretically ideal rectangular-shaped, renewal PSPs with biologically more realistic alpha-shaped, additive PSPs. Simulation results suggest that the network model with alpha-shaped PSPs does not capture the target distribution as accurately as with the theoretically ideal PSP shapes, statistical dependencies between the RVs $\mathbf{z}$ are however still approximated reasonably well.

5.3.4 Neural sampling in continuous time

The neural sampling model proposed above was formulated in discrete time of step size dt, inspired by the discrete time nature of MCMC techniques in statistics and machine learning as well as to make simulations possible on digital computers. However, models in continuous time (e.g., ordinary differential equations) are arguably more natural and "realistic" descriptions of temporally varying biological processes. This gives rise to the question whether one can find a sensible limit of the discrete time model in the limit $dt \to 0$, yielding a sampling network model in continuous time. Another motivation for considering continuous time models for neural sampling is the fact that many mathematical models for recurrent networks are formulated in continuous time [73], and a comparison to these existing models would be facilitated. Here we propose a stochastically spiking neural network model in continuous time, whose states still represent correct samples from the desired probability distribution $p(\mathbf{z})$ at any time t. These types of models are usually referred to as Markov jump processes. It can be shown that discretizing this continuous time model yields the discrete time model defined earlier, which thus can be regarded as a version suitable for simulations on a digital computer.

We define the continuous time model in the following way. Let t_k^l, for $l = 0, 1, \ldots$, denote the firing times of neuron ν_k. The refractory process of this neuron, in analogy to Figure 5.1 and equation (5.8)-(5.9) for the case of discrete time, is described by the following differential equation for the auxiliary variable ζ_k, which may now assume any nonnegative real number $0 \leq \zeta_k \leq 1$:

$$\frac{d}{dt}\zeta_k(t) = \begin{cases} -\frac{1}{\tau} & \text{for} \quad \zeta_k > 0 \\ \sum_l \delta(t - t_k^l) & \text{for} \quad \zeta_k = 0 \end{cases} . \tag{5.12}$$

Here $\delta(t - t_k^l)$ denotes Dirac's Delta centered at the spike time t_k^l. This differential equation describes the following simple dynamics. The auxiliary variable $\zeta_k(t)$ decays

linearly with time constant τ when the neuron is refractory, i.e., $\zeta_k(t) > 0$. Once $\zeta_k(t)$ arrives at its resting state 0 it remains there, corresponding to the neuron being ready to spike again (more precisely, in order to avoid point measures we set it to a random value in $[-2\epsilon, -\epsilon]$, see Methods). In the resting state, the neuron has the probability density $\frac{1}{\tau}\exp(u_k(t))$ to fire at every time t. If it fires at t_k^l, this results in setting $\zeta_k(t_k^l) = 1$, which is formalized in equation (5.12) by the sum of Dirac Delta's $\sum_l \delta(t - t_k^l)$. Here the current membrane potential $u_k(t)$ at time t is defined as in the discrete time case, e.g., by $u_k = b_k + \sum_{i=1}^K W_{ki} z_i(t)$ for the case of a Boltzmann distribution (5.5). The binary variable $z_k(t)$ is defined to be 1 if $\zeta_k(t) > 0$ and 0 if the neuron is in the resting state $\zeta_k(t) = 0$. Biologically, the term $W_{ki} z_i(t)$ can again be interpreted as the value at time t of a rectangular-shaped PSP (with a duration of τ) that neuron ν_i evokes in neuron ν_k. As the spikes are discrete events in continuous time, the probability of two or more neurons spiking at the same time is zero. This allows for updating all neurons in parallel using a differential equation.

In analogy to the discrete time case, the neural network in continuous time can be shown to sample from the desired distribution $p(\mathbf{z})$, i.e., $p(\mathbf{z})$ is an invariant distribution of the network dynamics defined above. However, to establish this fact, one has to rely on a different mathematical framework. The probability distribution $p_t(\zeta)$ of the auxiliary variables $\zeta_1(t), \ldots, \zeta_K(t)$ as a function of time t, which describes the evolution of the network, obeys a partial differential equation, the so-called Differential-Chapman-Kolmogorov equation (see [69]):

$$\partial_t p_t(\zeta) = (T p_t)(\zeta), \tag{5.13}$$

where the operator T, which captures the dynamics of the network, is implicitly defined by the differential equations (5.12) and the spiking probabilities. This operator T is the continuous time equivalent to the transition operator T in the discrete time case. The operator T consists here of two components. The *drift term* captures the deterministic decay process of $\zeta_k(t)$, stemming from the term $-1/\tau$ in equation (5.12). The *jump term* describes the non-continuous aspects of the path $\zeta_k(t)$ associated with "jumping" from $\zeta_k(t_k^l - dt) = 0$ to $\zeta_k(t_k^l) = 1$ at the time t_k^l when the neuron fires.

In the Methods section we prove that the resulting time invariant distribution, i.e., the distribution that solves $\partial_t p_t(\zeta) = 0$, now denoted $p(\zeta)$ as it is not a function of

time, gives rise to the desired marginal distribution $p(\mathbf{z})$ over $\mathbf{z}$:

$$\int d\zeta \; \delta(\mathbf{z}, \zeta^{>0}) \; p(\zeta) = p(\mathbf{z}) \; , \tag{5.14}$$

where $\delta(\mathbf{z}, \zeta^{>0}) = (\delta(z_1, \zeta_1^{>0}), \ldots, \delta(z_K, \zeta_K^{>0}))$ and $\zeta_k^{>0} = 1$ if $\zeta_k > 0$ and $\zeta_k^{>0} = 0$ otherwise. $\delta(z_k, \zeta_k^{>0}) = 1$ denotes Kronecker's Delta with $\delta(z_k, \zeta_k^{>0}) = 1$ if $z_k = \zeta_k^{>0}$ and $\delta(z_k, \zeta_k^{>0}) = 0$ otherwise. Thus, the function $\delta(\mathbf{z}, \zeta^{>0})$ simply reflects the definition that $z_k(t) = 1$ if $\zeta_k(t) > 0$ and 0 otherwise. For an explicit definition of T, a proof of the above statement, and some additional comments see the Methods section.

The neural samplers in discrete and continuous time are closely related. The model in discrete time provides an increasingly more precise description of the inherent spike dynamics when the duration dt of the discrete time step is reduced, causing an increase of τ (such that $\tau \cdot dt$ is constant) and therefore a reduced firing probability of each neuron at any discrete time step (see the term $\log \tau$ in equation (5.8)). In the limit of dt approaching 0, the probability that two or more neurons will fire at the same time approaches 0, and the discrete time sampler becomes equal to the continuous time system defined above, which updates all units in parallel.

It is also possible to formulate a continuous time version of the neural sampler based on neuron models with relative refractory mechanisms. In the Methods section the resulting continuous time neuron model with a relative refractory mechanism is defined. Theoretical results similar to the discrete time case can be derived for this sampler (see Lemmata 9 and 10 in Methods): It is shown that each neuron "locally" performs the correct computation under the assumption of static input from the remaining neurons. However one can no longer prove in general that the global network samples from the target distribution p.

5.3.5 Demonstration of probabilistic inference with recurrent networks of spiking neurons in an application to perceptual multistability

In the following we present a network model for perceptual multistability based on the neural sampling framework introduced above. This simulation study is aimed at showing that the proposed network can indeed sample from a desired distribution and also perform inference, i.e., sample from the correct corresponding posterior distribution. It is not meant to be a highly realistic or exhaustive model of perceptual multistability

nor of biologically plausible learning mechanisms. Such models would naturally require considerably more modelling work.

Perceptual multistability evoked by ambiguous sensory input, such as a 2D drawing (e.g., Necker cube) that allows for different consistent 3D interpretations, has become a frequently studied perceptual phenomenon. The most important finding is that the perceptual system of humans and nonhuman primates does not produce a superposition of different possible percepts of an ambiguous stimulus, but rather switches between different self-consistent global percepts in a spontaneous manner. Binocular rivalry, where different images are presented to the left and right eye, has become a standard experimental paradigm for studying this effect [4, 15, 23, 138]. A typical pair of stimuli are the two images shown in Figure 5.4A. Here the percepts of humans and nonhuman primates switch (seemingly stochastically) between the two presented orientations. [72, 104, 215] propose that several aspects of experimental data on perceptual multistability can be explained if one assumes that percepts correspond to samples from the conditional distribution over interpretations (e.g., different 3D shapes) given the visual input (e.g., the 2D drawing). Furthermore, the experimentally observed fact that percepts tend to be stable on the time scale of seconds suggests that perception can be interpreted as probabilistic inference that is carried out by MCMC sampling which produces successively correlated samples. In [72] it is shown that this MCMC interpretation is also able to qualitatively reproduce the experimentally observed distribution of dominance durations, i.e., the distribution of time intervals between perceptual switches. However, in lack of an adequate model for sampling by a recurrent network of spiking neurons, theses studies could describe this approach only on a rather abstract level, and pointed out the open problem to relate this algorithmic approach to neural processes. We have demonstrated in a computer simulation that the previously described model for neural sampling could in principle fill this gap, providing a modelling framework that is on the one hand consistent with the dynamics of networks of spiking neurons, and which can on the other hand also be clearly understood from the perspective of probabilistic inference through MCMC sampling.

In the following we model some essential aspects of an experimental setup for binocular rivalry with grating stimuli (see Figure 5.4A) in a recurrent network of spiking neurons with the previously described relative refractory mechanism. We assigned to each of the 217 neurons in the network $\mathcal{N}$ a tuning curve $V_k(\varphi)$, centered around its

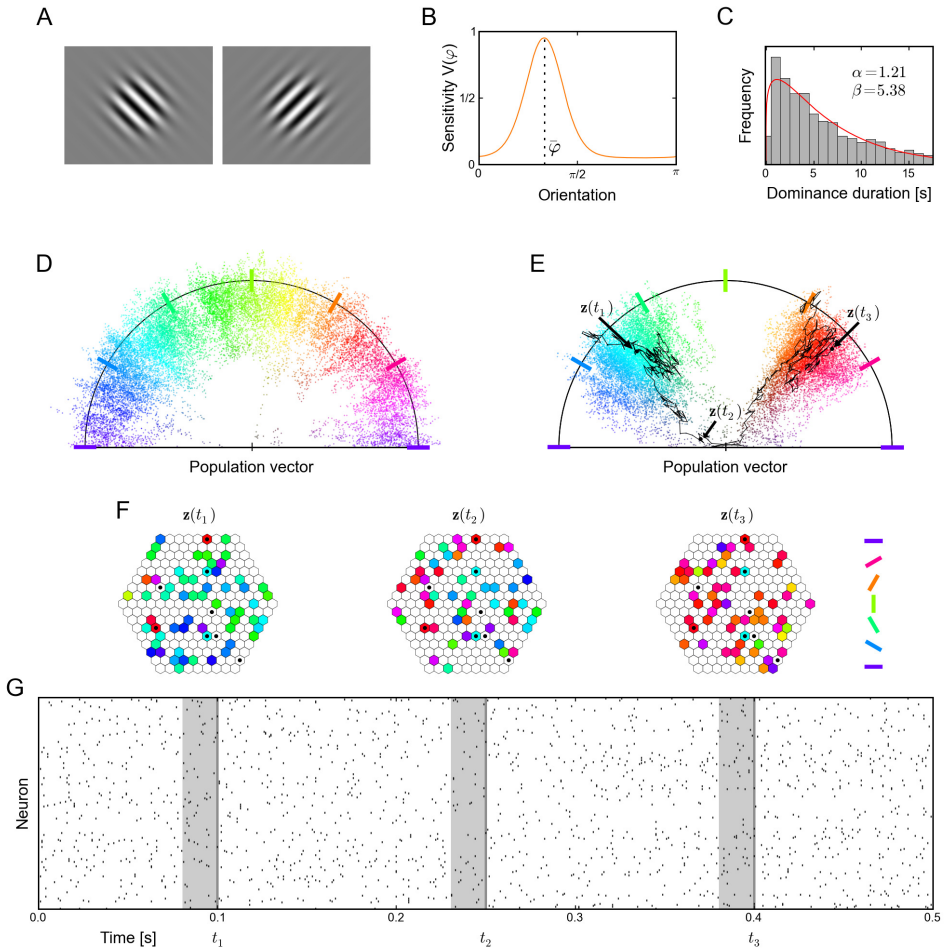

Figure 5.4: Modeling perceptual multistability as probabilistic inference with neural sampling. (A) Typical visual stimuli for the left and right eye in binocular rivalry experiments. (B) Tuning curve of a neuron with preferred orientation $\bar{\varphi}$. (C) Distribution of dominance durations in the trained network under ambiguous input. The red curve shows the Gamma distribution with maximum likelihood on the data. (D) 2-dimensional projection (via population vector) of the distribution $p(\mathbf{z})$ encoded in the spiking network showing that it strongly favors coherent global states of arbitrary orientation to incoherent ones (corresponding to population vectors of small magnitude). (E) 2-dimensional projection of the bimodal posterior distribution under an ambiguous input consisting of two different orientations reminiscent of the stimuli shown in A. The black trace shows the temporal evolution of the network state $\mathbf{z}$ for 500 ms around a perceptual switch. (F) Network states at 3 time points t_1, t_2, t_3 marked in E. Neurons that fired in the preceding 20 ms (see gray bar in G) are plotted in the color of their preferred orientation. Inactive neurons are shown in white. While states $\mathbf{z}(t_1)$ and $\mathbf{z}(t_3)$ represent rather coherent orientations, $\mathbf{z}(t_2)$ shows an incoherent state corresponding to a perceptual switch. Clamped neurons (which the posterior is condition on) are marked by a black dot. (G) Spike raster of the unclamped neurons during a 500 ms epoch marked by the black trace in E. Gray bars indicate the 20 ms time intervals that define the network states shown in F. Altogether this figure shows that a theoretically rigorous probabilistic inference process can be carried out by a network of spiking neurons with a spike raster that is similar to generic recorded data.

preferred orientation $\bar{\varphi}_k$ as shown in Figure 5.4B. The preferred orientations $\bar{\varphi}_k$ of the neurons were chosen to cover the entire interval $[0, \pi)$ of possible orientations and were randomly assigned to the neurons. The neurons were arranged on a hexagonal grid as depicted in Figure 5.4F. Any two neurons with distance ≤ 8 were synaptically connected (neighboring units had distance 1). We assume that these neurons represent neurons in the visual system that have roughly the same or neighboring receptive field, and that each neuron receives visual input from either the left or the right eye. The network connections were chosen such that neurons that have similar (very different) preferred orientations are connected with positive (negative) weights (for details see Methods section).

We examined the resulting distribution $p(\mathbf{z})$ over the 217 dimensional network states. To provide an intuitive visualization of these high dimensional network states $\mathbf{z}$, we resort to a 2-dimensional projection, the population vector of a state $\mathbf{z}$ (see Methods for details of the applied population vector decoding scheme). Only the endpoints of the population vectors are drawn (as colored points) in Figure 5.4D,E. The orientation of the population vector is assumed to correspond to the dominant orientation of the percept, and its distance from the origin encodes the strength of this percept. We also, somewhat informally, call the strength of a percept its coherence and a network state which represents a coherent percept a coherent network state. A coherent network state hence results in a population vector of large magnitude. Each direction of a population vector is color coded in Figure 5.4D,E, using the color code for directions shown on the right hand side of Figure 5.4F. In Figure 5.4D the distribution $p(\mathbf{z})$ of the network is illustrated by sampling of the network for 20 s, with samples $\mathbf{z}$ taken every millisecond. Each dot equals a sampled network state $\mathbf{z}$. In a biological interpretation the spike response of the freely evolving network reflects spontaneous activity, since no observations, i.e., no external input, was added to the system. Figure 5.4D shows that the spontaneous activity of this simple network of spiking neurons moves preferably through coherent network states for all possible orientations due to the chosen recurrent network connections (being positive for neurons with similar preferred orientation and negative otherwise). This can directly be seen from the rare occurrence of population vectors with small magnitude (vectors close to the "center") in Figure 5.4D.

To study percepts elicited by ambiguous stimuli, where inputs like in Figure 5.4A are shown simultaneously to the left and right eye during a binocular rivalry experiment,

we provided ambiguous input to the network. Two cells with preferred orientation $\bar{\varphi}_k \approx 45^\circ$ and two cells with $\bar{\varphi}_k \approx 135^\circ$ were clamped to 1. Additionally four neurons with $\bar{\varphi}_k \approx 0^\circ$ resp. 90° were muted by clamping to 0. This ambiguous input is incompatible with a coherent percept, as it corresponds to two orthogonal orientations presented at the same time. The resulting distribution over the state of the 209 remaining neurons is shown for a time span of 20 s of simulated biological time (with samples taken every millisecond) in Figure 5.4E. One clearly sees that the network spends most of the time in network states that correspond to one of the two simultaneously presented input orientations (45^o and 135^o), and virtually no time on orientations in between. This implements a sampling process from a bimodal conditional distribution. The black line marks a 500 ms trace of network states $\mathbf{z}$ around a perceptual switch: The network remained in one mode of high probability – corresponding to one percept – for some period of time, and then quickly traversed the state space to another mode – corresponding to a different percept.

Three of the states $\mathbf{z}$ around this perceptual switch ($\mathbf{z}(t_1)$, $\mathbf{z}(t_2)$ and $\mathbf{z}(t_3)$ in Figure 5.4E) are explicitly shown in Figure 5.4F. Neurons ν_k that fired during the preceding interval of 20 ms (marked in gray in Figure 5.4G) are drawn in the respective color of their preferred orientation. Inactive neurons are drawn in white, and clamped neurons are marked by a black dot (•).

Figure 5.4G shows the action potentials of the 209 non-clamped neurons during the same 500 ms trace around the perceptual switch. One sees that the sampling process is expressed in this neural network model by a sparse, asynchronous and irregular spike response. It is worth mentioning that the average firing rate when sampling from the posterior distribution is only slightly higher than the average firing rate of spontaneous activity (16.1 Hz and 15.4 Hz respectively), which is reminiscent of related experimental data [63]. Thus on the basis of the overall network activity it is indistinguishable whether the network carries out an inference task or freely samples from its prior distribution. It is furthermore notable, that a focus of the network activity on the two orientations that are given by the external input can be achieved in this model, in spite of the fact that only two of the 217 neurons were clamped for each of them. This numerical relationship is reminiscent of standard data on the weak input from LGN to V1 that is provided in the brain [19, 21], and raises the question whether the proposed neural sampling model could provide a possible mechanism (under the

modelling assumptions made above) for cortical processing of such numerically weak external inputs.

The distribution of the resulting dominance durations, i.e., the time between perceptual switches, for the previously described setup with ambiguous input is shown for a continuous run of 10^4 s in Figure 5.4C (a similar method as in [72] was used to measure dominance durations, see Methods). This distribution can be approximated quite well by a Gamma distribution, which also provides a good fit to experimental data (see the discussion in [72]). We expect that also other features of the more abstract MCMC model for biological vision of [72, 215], such as contextual biases and traveling waves, will emerge in larger and more detailed implementations of the MCMC approach through the proposed neural sampling method in networks of spiking neurons.

5.4 Discussion

We have presented a spiking neural network that samples from a given probability distribution via its inherent network dynamics. In particular the network is able to carry out probabilistic inference through sampling. The model, based on assumptions about the underlying probability distribution (formalized by the neural computability condition) as well as on certain assumptions regarding the underlying MCMC model, provides one possible neural implementation of the "inference-by-sampling paradigm" emerging in computational neuroscience.

During inference the observations (i.e., the variables which we wish to condition on) are modeled in this study by clamping the corresponding neurons by strong external input to the observed binary value. Units which receive no input or input with vanishing contrast (stimulus intensity) are treated as unobserved. Using this admittedly quite simplistic model of the input, we observed in simulations that our network model exhibits the following property: The onset of a sensory stimulus reduces the variability of the firing activity, which represents (after stimulus onset) a conditional distribution, rather than the prior distribution (see the difference between panels **D** and **E** of Figure 5.4). It is tempting to compare these results to the experimental finding of reduced firing rate variability after stimulus onset observed in several cortical areas [32]. We wish to point out however, that a consistent treatment of zero contrast stimuli

requires more thorough modelling efforts (e.g., by explicitly adding a random variable for the stimulus intensity [16, 62]), which is not the focus of the presented work.

Virtually all high-level computational tasks that a brain has to solve can be formalized as optimization problems, that take into account a (possibly large) number of soft or hard constraints. In typical applications of probabilistic inference in science and engineering (see e.g. [22, 125]) such constraints are encoded in e.g., conditional probability tables or factors. In a biological setup they could possibly be encoded through the synaptic weights of a recurrent network of spiking neurons. The solution of such optimizations problems in a probabilistic framework via sampling, as implemented in our model, provides an alternative to deterministic solutions, as traditionally implemented in neural networks (see, e.g., [103] for the case of constraint satisfaction problems). Whereas an attractor neural network converges to *one* (possibly approximate) solution of the problem, a stochastic network may alternate between different approximate solutions and stay the longest at those approximate solutions that provide the best fit. This might be advantageous, as given more time a stochastic network can explore more of the state space and avoid shallow local minima. Responses to ambiguous sensory stimuli [4, 15, 23, 138] might be interpreted as an optimization with soft constraints. The interpretation of human thinking as sampling process solving an inference task, recently proposed in cognitive science [47, 87, 228], further emphasizes that considering neural activity as an inferential process via sampling promises to be a fruitful approach.

Our approach builds on, and extends, previous work where recurrent networks of non-spiking stochastic neurons (commonly considered in artificial neural networks) were shown to be able to carry out probabilistic inference through Gibbs sampling [3]. In [102] a first extension of this approach to a network of recurrently connected spiking neurons had been presented. The dynamics of the recurrently connected spiking neurons are described as stepwise sampling from the posterior of a temporal Restricted Boltzmann Machine (tRBM) by introducing a clever interpretation of the temporal spike code as time varying parameters of a multivariate Gaussian distribution. Drawing one sample from the posterior of a RBM is, by construction, a trivial one-step task. In contrast to our model, the model of [102] does not produce multiple samples from a fixed posterior distribution, given the fixed input, but produces exactly one sample consisting of the temporal sequence of the hidden nodes, given a temporal input sequence. Similar temporal models, sometimes called Bayesian filtering, also underlie the

important contributions of [236] and [45]. In [45] every single neuron is described as hidden Markov Model (HMM) with two states. Instead of drawing samples from the instantaneous posterior distribution using stochastic spikes, [45] presents a deterministic spike generation with the intention to convey the analog probability value rather then discrete samples. The approach presented here can be interpreted as a biologically more realistic version of Gibbs sampling for a specific class of probability distributions by taking into account a spike-based communication, finite duration PSPs and refractory mechanisms. Other implementations based on different distributions (e.g., directed graphical models) and different sampling methods (e.g., reversible MCMC methods) are of course conceivable and worth exploring.

In a computer experiment (see Figure 5.4), we used our proposed network to model aspects of biological vision as probabilistic inference along the lines of argumentation put forward in [72, 104, 215]. Our model was chosen to be quite simplistic, just to demonstrate that a number of experimental data on the dynamics of spontaneous activity [16, 65, 119] and binocular rivalry [4, 15, 23, 138] can in principle be captured by this approach. The main point of the modelling study is to show that rather realistic neural dynamics can support computational functions rigorously formalized as inference via sampling.

We have also presented a model of spiking dynamics in continuous time that performs sampling from a given probability distribution. Although computer simulations of biological networks of neurons often actually use discrete time, it is desirable to also have a sound approach for understanding and describing the network sampling dynamics in continuous time, as the latter is arguable a natural framework for describing temporal processes in biology. Furthermore comparison to many existing continuous time neuron and network models of neurons is facilitated.

We have made various simplifying assumption regarding neural processes, e.g., simple symbolic postsynaptic potentials in the form of step-functions (reminiscent of plateau potentials caused by dendritic NMDA spikes [7]). More accurate models for neurons have to integrate a multitude of time constants that represent different temporal processes on the physical, molecular, and genetic level. Hence the open problem arises, to which extent this multitude of time constants and other complex dynamics can be integrated into theoretical models of neural sampling. We have gone one first step in this direction by showing that in computer simulations the two temporal

processes that we have considered (refractory processes and postsynaptic potentials) can approximately be decoupled. Furthermore, we have presented simulation results suggesting that more realistic alpha-shaped, additive EPSPs are compatible with the functionality of the proposed network model.

Finally, we want to point out that the prospect of using networks of spiking neurons for probabilistic inference via sampling suggests new applications for energy-efficient spike-based and massively parallel electronic hardware that is currently under development [26, 151].

5.5 Methods

We first provide details and proofs for the neural sampling models, followed by details for the computer simulations. Then we investigate typical firing statistics of individual neurons during neural sampling and examine the approximation quality of neural sampling with different neuron and synapse models.

5.5.1 Mathematical details

5.5.1.1 Notation

To keep the derivations in a compact form, we introduce the following notations. We define the function $\zeta_k^{>0}$ of ζ_k to be 1 if $\zeta_k > 0$ and 0 otherwise. Analogously we define $\zeta_{\setminus k}^{>0} = (\zeta_1^{>0}, \ldots, \zeta_{k-1}^{>0}, \zeta_{k+1}^{>0}, \ldots, \zeta_K^{>0})$. Let $\delta(\cdot,\cdot)$ denote Kronecker's Delta, i.e., $\delta(x,y) = 1$ if $x = y$ and 0 whereas $\delta(\cdot)$ denotes Dirac's Delta, i.e., $\int f(x)\delta(x)dx = f(0)$. Furthermore $\chi_I(x)$ is the indicator function of the set I, i.e., $\chi_I(x) = 1$ if $x \in I$ and $\chi_I(x) = 0$ if $x \notin I$.

5.5.1.2 Details to neural sampling with absolute refractory period in discrete time

The following Lemmata 1 – 3 provide a proof of Theorem 1. For completeness we begin this paragraph with a recapitulation of the definitions stated in Results. We then identify some central properties of the joint probability distribution $p(\zeta, \mathbf{z})$ and proof that the proposed network samples from the desired invariant distribution.

For a given distribution $p(\mathbf{z})$ over the binary variables $\mathbf{z} \in \{0,1\}^K$ with $\forall \mathbf{z} \in \{0,1\}^K$ $p(\mathbf{z}) \neq 0$, the joint distribution over $(\zeta, \mathbf{z})$ with $\zeta \in \{0, 1, \ldots, \tau\}^K$ is defined in

the following way (see equation 5.7):

$$\begin{aligned} p(\zeta_k|z_k) &:= \begin{cases} \tau^{-1} & \text{for} \quad z_k = 1 \ \wedge \ \zeta_k > 0 \\ 1 & \text{for} \quad z_k = 0 \ \wedge \ \zeta_k = 0 \\ 0 & \text{otherwise} \end{cases} \\ p(\zeta|\mathbf{z}) &:= \prod_{k=1}^{K} p(\zeta_k|z_k) \\ p(\zeta, \mathbf{z}) &:= p(\zeta|\mathbf{z})p(\mathbf{z}). \end{aligned}$$

The assumption $p(\mathbf{z}) \neq 0$ for all $\mathbf{z}$ is required to show the irreducibility of the Markov chain, a prerequisite to ensure the uniqueness of the invariant distribution of the MCMC dynamics. Furthermore, for the given distribution $p(\mathbf{z})$ we define the functions $u_k : \{0,1\}^{K-1} \to \mathbb{R}$ for $k \in \{1, \ldots, K\}$ which map $\mathbf{z}_{\setminus k} \mapsto u_k(\mathbf{z}_{\setminus k})$:

$$u_k(\mathbf{z}_{\setminus k}) := \operatorname{logit}(p(z_k = 1|\mathbf{z}_{\setminus k})) = \log \frac{p(z_k = 1|\mathbf{z}_{\setminus k})}{p(z_k = 0|\mathbf{z}_{\setminus k})}.$$

Instead of $u_k(\mathbf{z}_{\setminus k})$ we simply write u_k in the following.

Lemma 1. *The distribution $p(\zeta, \mathbf{z})$ has conditional distributions of the following form:*

$$\begin{aligned} p(\zeta_k|\zeta_{\setminus k}, \mathbf{z}_{\setminus k}) = p(\zeta_k|\mathbf{z}_{\setminus k}) &= \begin{cases} \frac{\sigma(u_k)}{\tau} & \text{for} \quad \zeta_k > 0 \\ 1 - \sigma(u_k) & \text{otherwise} \end{cases} \\ p(z_k|\zeta, \mathbf{z}_{\setminus k}) = p(z_k|\zeta_k) &= \begin{cases} 1 & \text{for} \quad \zeta_k > 0 \ \wedge z_k = 1 \\ 1 & \text{for} \quad \zeta_k = 0 \ \wedge z_k = 0 \\ 0 & \text{otherwise} \ . \end{cases} \end{aligned}$$

These results can also be written more compactly in the following form: $p(\zeta_k|\mathbf{z}_{\setminus k}) = \sigma(u_k)\chi_{\{1,\ldots,\tau\}}(\zeta_k)\frac{1}{\tau} + (1 - \sigma(u_k))\delta(\zeta_k, 0)$ and $p(z_k|\zeta_k) = \delta(z_k, \zeta_k^{>0})$.

Proof. Here we use the fact that the logistic function σ is the inverse of the logit function, i.e., $p(z_k = 1|\mathbf{z}_{\setminus k}) = \sigma(u_k)$.

$$\begin{aligned} p(\zeta_k|\zeta_{\setminus k}, \mathbf{z}_{\setminus k}) &= \sum_{z_k=0}^{1} \frac{p(\zeta, \mathbf{z})}{p(\zeta_{\setminus k}, \mathbf{z}_{\setminus k})} = \sum_{z_k=0}^{1} \frac{p(\zeta, \mathbf{z})}{p(\zeta_{\setminus k}|\mathbf{z}_{\setminus k})p(\mathbf{z}_{\setminus k})} = \sum_{z_k=0}^{1} \frac{\left(\prod_{l\neq k} p(\zeta_l|z_l)\right) p(\zeta_k|z_k)p(\mathbf{z})}{\left(\prod_{l\neq k} p(\zeta_l|z_l)\right) p(\mathbf{z}_{\setminus k})} \\ &= \sum_{z_k=0}^{1} p(\zeta_k|z_k)p(z_k|\mathbf{z}_{\setminus k}) = \sigma(u_k)\chi_{\{1,\ldots,\tau\}}(\zeta_k)\frac{1}{\tau} + (1 - \sigma(u_k))\delta(\zeta_k, 0). \end{aligned}$$

This also shows that ζ_k is independent from $\zeta_{\setminus k}$ given $\mathbf{z}_{\setminus k}$, i.e, $p(\zeta_k|\zeta_{\setminus k}, \mathbf{z}_{\setminus k}) = p(\zeta_k|\mathbf{z}_{\setminus k})$. Now we show the second relation using Bayes' rule:

$$\begin{aligned} p(z_k|\zeta, \mathbf{z}_{\setminus k}) &= \frac{p(\zeta_k|\zeta_{\setminus k}, \mathbf{z})}{p(\zeta_k|\zeta_{\setminus k}, \mathbf{z}_{\setminus k})} p(z_k|\zeta_{\setminus k}, \mathbf{z}_{\setminus k}) \\ &= \frac{z_k \chi_{\{1,\ldots,\tau\}}(\zeta_k)\frac{1}{\tau} + (1-z_k)\delta(\zeta_k, 0)}{\sigma(u_k)\chi_{\{1,\ldots,\tau\}}(\zeta_k)\frac{1}{\tau} + (1-\sigma(u_k))\delta(\zeta_k, 0)} p(z_k|\mathbf{z}_{\setminus k}) \\ &= \begin{cases} z_k & \text{for} \quad \zeta_k > 0 \\ 1 - z_k & \text{for} \quad \zeta_k = 0 \end{cases} \\ &= \delta(z_k, \zeta_k^{>0}). \end{aligned}$$

□

In order to facilitate the verification of the next two Lemmata, we first restate the definition of the operators T^k in a more concise way:

$$\begin{aligned} T &:= T^1 \circ \ldots \circ T^K \\ T^k(\zeta, \mathbf{z}|\zeta', \mathbf{z}') &:= T^k(\zeta_k, z_k|\zeta', \mathbf{z}')\delta(\zeta_{\setminus k}, \zeta'_{\setminus k})\delta(\mathbf{z}_{\setminus k}, \mathbf{z}'_{\setminus k}) \\ T^k(\zeta_k, z_k|\zeta', \mathbf{z}') &:= \delta(z_k, \zeta_k^{>0}) \cdot T^k(\zeta_k|\zeta'_k, \mathbf{z}'_{\setminus k}) \\ T^k(\zeta_k|\zeta'_k, \mathbf{z}'_{\setminus k}) &:= \begin{cases} \sigma(u'_k - \log\tau) & \text{for} & \zeta_k = \tau \ \wedge \ \zeta'_k = 0, 1 \\ 1 - \sigma(u'_k - \log\tau) & \text{for} & \zeta_k = 0 \ \wedge \ \zeta'_k = 0, 1 \\ 1 & \text{for} & \zeta_k = \zeta'_k - 1 \ \wedge \ \zeta'_k > 1 \\ 0 & \text{otherwise} & \end{cases}, \end{aligned}$$

where $u'_k := u_k(\mathbf{z}'_{\setminus k}) = \text{logit}(p(z_k = 1|\mathbf{z}'_{\setminus k}))$.

Lemma 2. *For all $k = 1, \ldots, K$ the operator $T^k(\zeta_k|\zeta'_k, \mathbf{z}'_{\setminus k})$ leaves the conditional distribution $p(\zeta_k|\mathbf{z}'_{\setminus k})$ invariant.*

Proof. For sake of simplicity, denote $T^k(\zeta_k = i|\zeta'_k = j, \mathbf{z}'_{\setminus k}) = T^k_{ij}$ for $i, j \in \{0, 1, \ldots, \tau\}$ and $p(\zeta_k = i|\mathbf{z}'_{\setminus k}) = p_i$. We have to show $p_i \stackrel{!}{=} \sum_{j=0}^{\tau} T^k_{ij} p_j$ for $i \in \{0, 1, \ldots, \tau\}$.

First we show $p_\tau = \sum_{j=0}^{\tau} T^k_{\tau j} p_j$ using $p_0 = 1 - \sigma(u_k)$ and $p_1 = p_2 = \ldots = p_\tau = \sigma(u_k)\tau^{-1}$ (which results from Lemma 1):

$$\begin{aligned} \sum_{j=0}^{\tau} T^k_{\tau j} p_j &= T^k_{\tau 0} p_0 + T^k_{\tau 1} p_1 = \sigma(u_k - \log\tau)(1 - \sigma(u_k)) + \sigma(u_k - \log\tau)\sigma(u_k)\tau^{-1} \\ &= \sigma(u_k - \log\tau)\sigma(u_k)\tau^{-1}\left(\tau\exp(-u_k) + 1\right) = \sigma(u_k - \log\tau)\sigma(u_k)\tau^{-1}(\sigma(u_k - \log\tau))^{-1} \\ &= \sigma(u_k)\tau^{-1} \stackrel{!}{=} p_\tau. \end{aligned}$$

Here we used the definition of the logistic function $\sigma(x) = (1+\exp(-x))^{-1}$ and $\sigma(x)(1-\sigma(x))^{-1} = \exp(x)$.

Now we show $p_0 = \sum_{j=0}^{\tau} T_{0j}^k p_j$:

$$\begin{aligned}
\sum_{j=0}^{\tau} T_{0j}^k p_j &= T_{00}^k p_0 + T_{01}^k p_1 \\
&= (1-\sigma(u_k - \log\tau))(1-\sigma(u_k)) + (1-\sigma(u_k-\log\tau))\sigma(u_k)\tau^{-1} \\
&= (1-\sigma(u_k-\log\tau))(1-\sigma(u_k))\left(1+\exp(u_k)\tau^{-1}\right) \\
&= \sigma(-u_k+\log\tau)(1-\sigma(u_k))(\sigma(-u_k+\log\tau))^{-1} \\
&= 1-\sigma(u_k) \stackrel{!}{=} p_0.
\end{aligned}$$

Here we used $1-\sigma(x) = \sigma(-x)$.

It is trivial to show $p_i = \sum_{j=0}^{\tau} T_{ij}^k p_j$ for $i = 1, \ldots, \tau-1$ as $\sum_{j=0}^{\tau} T_{ij}^k p_j = T_{i,i+1}^k p_{i+1} = p_{i+1} = p_i$. Here we used the facts that $T_{i,i+1}^k = 1$ and $p_i = p_{i+1}$ for $i = 1, \ldots, \tau-1$ by definition. □

Lemma 3. *For all $k = 1, \ldots, K$ the operator $T^k(\mathbf{z}, \zeta | \zeta', \mathbf{z}')$ leaves the distribution $p(\zeta, \mathbf{z})$ invariant.*

Proof. We start from Lemma 2, which states that $T^k(\zeta_k | \zeta'_k, \mathbf{z}'_{\setminus k})$ leaves the conditional

distribution $p(\zeta_k|\mathbf{z}'_{\setminus k})$ invariant:

$$\begin{aligned}
&\sum_{\zeta'_k} T^k(\zeta_k|\zeta'_k, \mathbf{z}'_{\setminus k}) p(\zeta'_k|\mathbf{z}'_{\setminus k}) = p(\zeta_k|\mathbf{z}'_{\setminus k}) \\
\Leftrightarrow &\sum_{\zeta'_k, \mathbf{z}'_{\setminus k}} \delta(\mathbf{z}_{\setminus k}, \mathbf{z}'_{\setminus k}) T^k(\zeta_k|\zeta'_k, \mathbf{z}'_{\setminus k}) p(\zeta'_k|\mathbf{z}'_{\setminus k}) = \sum_{\mathbf{z}'_{\setminus k}} \delta(\mathbf{z}_{\setminus k}, \mathbf{z}'_{\setminus k}) p(\zeta_k|\mathbf{z}'_{\setminus k}) = p(\zeta_k|\mathbf{z}_{\setminus k}) \\
\Leftrightarrow &\sum_{\zeta'_k, \mathbf{z}'_{\setminus k}} \delta(z_k, \zeta_k^{>0}) \delta(\mathbf{z}_{\setminus k}, \mathbf{z}'_{\setminus k}) T^k(\zeta_k|\zeta'_k, \mathbf{z}'_{\setminus k}) p(\zeta'_k|\mathbf{z}'_{\setminus k}) = \delta(z_k, \zeta_k^{>0}) p(\zeta_k|\mathbf{z}_{\setminus k}) = p(z_k|\zeta_k) p(\zeta_k|\mathbf{z}_{\setminus k}) \\
\Leftrightarrow &\sum_{\zeta'_k, \mathbf{z}'} \delta(z_k, \zeta_k^{>0}) \delta(\mathbf{z}_{\setminus k}, \mathbf{z}'_{\setminus k}) T^k(\zeta_k|\zeta'_k, \mathbf{z}'_{\setminus k}) p(z'_k, \zeta'_k|\mathbf{z}'_{\setminus k}) = p(z_k, \zeta_k|\mathbf{z}_{\setminus k}) \\
\Leftrightarrow &\sum_{\zeta'_k, \mathbf{z}'} \delta(z_k, \zeta_k^{>0}) \delta(\mathbf{z}_{\setminus k}, \mathbf{z}'_{\setminus k}) T^k(\zeta_k|\zeta'_k, \mathbf{z}'_{\setminus k}) p(z'_k, \zeta'_k|\mathbf{z}'_{\setminus k}) p(\zeta_{\setminus k}|\mathbf{z}'_{\setminus k}) p(\mathbf{z}'_{\setminus k}) \\
&\qquad\qquad = p(z_k, \zeta_k|\mathbf{z}_{\setminus k}) p(\zeta_{\setminus k}|\mathbf{z}_{\setminus k}) p(\mathbf{z}_{\setminus k}) \\
\Leftrightarrow &\sum_{\zeta'_k, \mathbf{z}'} \delta(z_k, \zeta_k^{>0}) \delta(\mathbf{z}_{\setminus k}, \mathbf{z}'_{\setminus k}) T^k(\zeta_k|\zeta'_k, \mathbf{z}'_{\setminus k}) p(\zeta'_k, \zeta_{\setminus k}, \mathbf{z}') = p(\zeta, \mathbf{z}) \\
\Leftrightarrow &\sum_{\zeta', \mathbf{z}'} T^k(z_k, \zeta_k|\zeta', \mathbf{z}') \delta(\zeta_{\setminus k}, \zeta'_{\setminus k}) \delta(\mathbf{z}_{\setminus k}, \mathbf{z}'_{\setminus k}) p(\mathbf{z}', \zeta') = p(\zeta, \mathbf{z}) \\
\Leftrightarrow &\sum_{\zeta', \mathbf{z}'} T^k(\mathbf{z}, \zeta|\zeta', \mathbf{z}') p(\mathbf{z}', \zeta') = p(\zeta, \mathbf{z}).
\end{aligned}$$

Here we used the relations $\delta(z_k, \zeta_k^{>0}) = p(z_k|\zeta_k)$ and $p(\zeta_k, z_k|\mathbf{z}_{\setminus k}) = p(z_k|\zeta_k) p(\zeta_k|\mathbf{z}_{\setminus k})$ as well as $p(\zeta_k|\mathbf{z}_{\setminus k}) = p(\zeta_k|\zeta_{\setminus k}, \mathbf{z}_{\setminus k})$ which directly follow from the definitions of $T^k(\zeta, \mathbf{z}, |\zeta', \mathbf{z}')$ and $p(\zeta, \mathbf{z})$. □

Finally, we can verify that the composed operator $T = T^1 \circ \ldots \circ T^K$ samples from the given distribution p.

Theorem 1. *$p(\zeta, \mathbf{z})$ is the unique invariant distribution of operator T.*

Proof. As all T^k leave $p(\zeta, \mathbf{z})$ invariant, so does the concatenation $T = T^1 \circ \ldots \circ T^K$. To ensure that $p(\zeta, \mathbf{z})$ is the *unique* invariant distribution, we have to show that T is irreducible and aperiodic. T is aperiodic as the transition probabilities $T^k_{00} = 1 - \sigma(u_k - \log \tau) > 0$ and $T^k_{00} < 1$ (this follows from the assumption $\forall \mathbf{z}\ p(\mathbf{z}) \neq 0$ made above).

The operator T is also irreducible for the following reason. First we see that from any state $(\zeta', \mathbf{z}')$ in at most τ steps we can get to the zero-state $(\zeta, \mathbf{z}) = 0^{2K}$ (and stay there) with non-zero probability, as $T^k_{i,i+1} = 1$ for $i = 1, \ldots, \tau - 1$ and $T^k_{01} = 1 - \sigma(u_k - \log \tau) > 0$. Furthermore, it can be seen that any state $(\hat{\zeta}, \hat{\mathbf{z}})$ can be reached

from the zero-state $(\zeta, \mathbf{z}) = 0^{2K}$ in at most τ steps since $T^k_{N0} = \sigma(u_k - \log\tau) > 0$ for any value of u_k. Hence every final state $(\hat{\zeta}, \hat{\mathbf{z}})$ can be reached from every starting state $(\zeta', \mathbf{z}')$ in at most 2τ steps with non-vanishing probability. □

5.5.1.3 Details to neural sampling with a relative refractory period in discrete time

We augment the neuron model with a relative refractory period described by a function $g(\zeta_k)$. We first ensure existence of the corresponding function $f(u_k)$. Based on these functions we then introduce the transition operator T of the Markov chain. This operator is shown to entail correct "local" computations.

Lemma 4. *Let $(g_1, \ldots, g_\tau) \in (\mathbb{R}^+_0)^\tau$ be a tuple of non-negative real numbers, with $g_\tau = 0$ and at least one element $g_i \geq 1$. This defines the refractory function via $g(\zeta_k) := g_{\zeta_k}$. There exists a unique $\mathcal{C}^\infty$ function $f : \mathbb{R} \to (0,1)$ with the following property $\forall u \in \mathbb{R}$:*

$$f(u)\frac{\sum_{i=1}^{\tau}\prod_{j=i+1}^{\tau}(1-g_j f(u))}{\prod_{j=1}^{\tau}(1-g_j f(u))} \quad = \quad \exp(u). \tag{5.15}$$

Furthermore, the function f has the property:

$$\begin{aligned} &\forall i \in \{1,\ldots,\tau\}\ \forall u \in \mathbb{R}: && 0 \leq g_i f(u) < 1 \\ &\exists i \in \{1,\ldots,\tau\}\ \forall u \in \mathbb{R}: && 0 < g_i f(u) < 1. \end{aligned}$$

Proof. Let $g_{\max} := \max_{j\in\{1,\ldots\tau\}} g_j$; we know that $g_{\max} \geq 1$. We define the function $F : (0, 1/g_{\max}) \to \mathbb{R}^+$:

$$F(x) \quad := \quad x\sum_{i=1}^{\tau}\left(\frac{1}{\prod_{j=1}^{i}(1-g_j x)}\right)$$

We can see that F is a positive $\mathcal{C}^\infty$ function on $(0, 1/g_{\max})$. Furthermore, $F(x)/x$ is defined as a sum of functions of the form $\frac{1}{\prod_{j=1}^{i}(1-g_j x)}$. Each factor $1/(1-g_j x)$ is positive and strictly monotonous. Therefore, F is strictly monotonous on $(0, 1/g_{\max})$ with the limits:

$$\begin{aligned} \lim_{x\to 0} F(x) &= 0 \\ \lim_{x\to 1/g_{\max}} F(x) &= \infty. \end{aligned}$$

Hence the equation $F(x) = \exp(u)$ has a unique solution for x called $f(u) \in (0, 1/g_{\max})$ for all $u \in \mathbb{R}$. From applying the implicit function theorem to $F(x,u) := F(x) - \exp(u)$ it follows that f is $\mathcal{C}^\infty$. □

From here on, with the letter f we will denote the function characterized by the above Lemma for the given tuple g (which denotes the chosen refractory function).

Definition 1. *Define $g_0 = 1$. The transition operator T^k is defined in the following way for all $k = 1, \ldots, K$:*

$$
\begin{aligned}
T^k(\zeta_k, z_k | \zeta', \mathbf{z}') &:= \delta(z_k, \zeta_k^{>0}) T^k(\zeta_k | \zeta'_k, \mathbf{z}'_{\setminus k}) \\
T^k(\zeta_k | \zeta'_k, \mathbf{z}'_{\setminus k}) &:= \begin{cases} g_{\zeta'_k} f(u_k) & \text{for} \quad \zeta_k = \tau \\ 1 - g_{\zeta'_k} f(u_k) & \text{for} \quad \zeta_k = \zeta'_k - 1 \ \wedge \ \zeta'_k > 0 \\ 1 - f(u_k) & \text{for} \quad \zeta_k = 0 \ \wedge \ \zeta'_k = 0 \\ 0 & \text{otherwise} \end{cases},
\end{aligned}
$$

with $u_k = u_k(\mathbf{z}'_{\setminus k})$.

Lemma 5. *For all $k = 1, \ldots, K$ the unique invariant distribution $q^*(z_k, \zeta_k | \zeta'_{\setminus k}, \mathbf{z}'_{\setminus k})$ of the operator $T^k(z_k, \zeta_k | \zeta', \mathbf{z}')$ fulfills $\sum_{\zeta_k} q^*(z_k, \zeta_k | \zeta'_{\setminus k}, \mathbf{z}'_{\setminus k}) = p(z_k | \mathbf{z}'_{\setminus k})$. This means, for a constant configuration $\mathbf{z}'_{\setminus k}$, the operator T^k produces samples z_k^* from the correct conditional distribution $p(z_k | \mathbf{z}'_{\setminus k})$.*

Proof. We define:

$$
\begin{aligned}
q^*(z_k, \zeta_k | \zeta'_{\setminus k}, \mathbf{z}'_{\setminus k}) &:= \delta(z_k, \zeta_k^{>0}) q(\zeta_k | \mathbf{z}'_{\setminus k}) \\
&:= \delta(z_k, \zeta_k^{>0}) \left(\sigma(u_k) h(\zeta_k | \mathbf{z}'_{\setminus k}) + (1 - \sigma(u_k)) \delta(\zeta_k, 0) \right),
\end{aligned}
$$

where the function $h(\zeta_k | \mathbf{z}'_{\setminus k})$ is defined as:

$$
h(\zeta_k | \mathbf{z}'_{\setminus k}) := \begin{cases} \frac{\prod_{j=\zeta_k+1}^{\tau} (1 - g_j f(u_k))}{\sum_{\alpha=1}^{\tau} \prod_{j=\alpha+1}^{\tau} (1 - g_j f(u_k))} & \text{for} \quad \zeta_k > 0 \\ 0 & \text{otherwise} \end{cases}.
$$

It is trivial to see that q^* has the correct marginal distribution over z_k:

$$
\begin{aligned}
\sum_{\zeta_k} q^*(z_k, \zeta_k | \zeta'_{\setminus k}, \mathbf{z}'_{\setminus k}) &= \sum_{\zeta_k} \delta(z_k, \zeta_k^{>0}) \left(\sigma(u_k) h(\zeta_k | \mathbf{z}'_{\setminus k}) + (1 - \sigma(u_k)) \delta(\zeta_k, 0) \right) \\
&= \sigma(u_k)^{z_k} (1 - \sigma(u_k))^{1 - z_k} = p(z_k | \mathbf{z}'_{\setminus k}).
\end{aligned}
$$

We now show that q^* is the unique invariant distribution of T^k. Because of the definition of T^k, we only have to show that $q^*(\zeta_k|\mathbf{z}'_{\setminus k})$ is the unique invariant distribution of $T^k(\zeta_k|\zeta'_k, \mathbf{z}'_{\setminus k})$. We denote $q^*(\zeta_k = i|\mathbf{z}'_{\setminus k}) =: q_i$ and $T^k(\zeta_k = i|\zeta'_k = j, \mathbf{z}'_{\setminus k}) =: T_{ij}$, i.e., we have to show $\forall i \in \{0, 1, \ldots, \tau\} : \; q_i = \sum_j T_{ij} q_j$.

It is trivial to show $q_i = \sum_j T_{ij} q_j$ for $1 \leq i \leq \tau - 1$, as there is only one non-vanishing element of transition operator, namely $T_{i,i+1}$:

$$\begin{aligned}
\sum_{j=0}^{\tau} T_{ij} q_j &= T_{i,i+1} q_{i+1} = (1 - g_{i+1} f(u_k)) q_{i+1} \\
&= (1 - g_{i+1} f(u_k)) h(\zeta_k = i + 1|\mathbf{z}_{\setminus k}) \sigma(u_k) \\
&= h(\zeta_k = i|\mathbf{z}_{\setminus k}) p(z_k = 1|\mathbf{z}_{\setminus k}) \stackrel{!}{=} q_i.
\end{aligned}$$

Here we used $q_i = h(\zeta_k = i|\mathbf{z}_{\setminus k})\sigma(u_k)$ for $i > 0$ and the definition of $h(\zeta_k|\mathbf{z}_{\setminus k})$.

Now we show $q_0 = \sum_j T_{0j} q_j$ starting from equation (5.15) and additionally using the relations $\exp(u_k) = \sigma(u_k)/(1 - \sigma(u_k))$ and $q_0 = 1 - \sigma(u_k)$ as well as the definition of q_1. We define for the sake of simplicity $\psi := \sum_{\alpha=1}^{\tau} \prod_{j=\alpha+1}^{\tau} (1 - g_j f(u_k))$:

$$\begin{aligned}
\sum_{j=0}^{\tau} T_{0j} q_j &= (1 - f(u_k)) q_0 + (1 - g_1 f(u_k)) q_1 \\
&= (1 - f(u_k))(1 - \sigma(u_k)) + \frac{\sigma(u_k)}{\psi} \prod_{j=1}^{\tau} (1 - g_j f(u_k)) \\
&= (1 - f(u_k))(1 - \sigma(u_k)) + \sigma(u_k) f(u_k) \exp(-u_k) \\
&= (1 - f(u_k))(1 - \sigma(u_k)) + f(u_k)(1 - \sigma(u_k)) \stackrel{!}{=} q_0.
\end{aligned}$$

We finally show $q_\tau = \sum_j T_{\tau j} q_j$, using the definition of $q_\tau = \sigma(u_k) h(\zeta_k = \tau | \mathbf{z}_{\setminus k}) = \frac{\sigma(u_k)}{\psi}$:

$$\begin{aligned}
\sum_{i=0}^{\tau} T_{\tau i} q_i &= \sum_{i=1}^{\tau} g_i f(u_k) q_i + f(u_k) q_0 \\
&= \sum_{i=1}^{\tau} g_i f(u_k) \prod_{j=i+1}^{\tau} (1 - g_j f(u_k)) \frac{\sigma(u_k)}{\psi} + f(u_k) q_0 \\
&= \frac{\sigma(u_k)}{\psi} \left(\sum_{i=1}^{\tau} g_i f(u_k) \prod_{j=i+1}^{\tau} (1 - g_j f(u_k)) + (1 - g_1 f(u_k)) \prod_{j=2}^{\tau} (1 - g_j f(u_k)) \right) \\
&= \frac{\sigma(u_k)}{\psi} \left(-\sum_{i=1}^{\tau} (1 - g_i f(u_k)) \prod_{j=i+1}^{\tau} (1 - g_j f(u_k)) + \sum_{i=1}^{\tau} \prod_{j=i+1}^{\tau} (1 - g_j f(u_k)) \right. \\
&\qquad \left. + \prod_{j=1}^{\tau} (1 - g_j f(u_k)) \right) \\
&= \frac{\sigma(u_k)}{\psi} \left(-\sum_{i=1}^{\tau} \prod_{j=i}^{\tau} (1 - g_j f(u_k)) + \sum_{i=0}^{\tau} \prod_{j=i+1}^{\tau} (1 - g_j f(u_k)) \right) \\
&= \frac{\sigma(u_k)}{\psi} \left(-\sum_{i=0}^{\tau-1} \prod_{j=i+1}^{\tau} (1 - g_j f(u_k)) + \sum_{i=0}^{\tau} \prod_{j=i+1}^{\tau} (1 - g_j f(u_k)) \right) \\
&= \frac{\sigma(u_k)}{\psi} \left(\prod_{j=\tau+1}^{\tau} (1 - g_j f(u_k)) \right) = \frac{\sigma(u_k)}{\psi} \stackrel{!}{=} q_\tau.
\end{aligned}$$

The argument that the transition operator T^k is aperiodic and irreducible is similar to the one presented in Lemma 1. □

5.5.1.4 Details to neural sampling with an absolute refractory period in continuous time

In contrast to the discrete time model we define the state space of ζ_k to be $\mathbb{R}^+ \cup [-2\epsilon, -\epsilon]$ for $\epsilon > 0$, i.e., as the union of the positive real numbers and a small interval $[-2\epsilon, -\epsilon]$. We will define the sampling operator in such a way that after neuron k was refractory for exactly its refractory period τ, its refractory variable ζ_k is uniformly placed in the small interval $[-2\epsilon, -\epsilon]$, which represents now the resting state and replaces $\zeta_k = 0$. This avoids point measures (Dirac's Delta) on the value $\zeta_k = 0$. This system is still exactly equivalent to the system discussed in the main paper, as all spike-transition probabilities of T for $\zeta_k < 0$ are constant. Hence, it does not matter which values ζ_k

assumes with respect to the spike mechanism during its non-refractory period as long as $\zeta_k < 0$.

Definition 2. *For a given distribution $p(\mathbf{z})$ over the binary variables $\mathbf{z} \in \{0,1\}^K$ with $\forall \mathbf{z} \in \{0,1\}^K \; p(\mathbf{z}) \neq 0$, we define a joint distribution over $(\zeta, \mathbf{z})$ with $\zeta \in \mathbb{R}^K$ in the following way:*

$$\begin{aligned} p(\zeta_k|z_k) &:= \begin{cases} 1 & \text{for} \quad 1 \geq \zeta_k > 0 \;\wedge\; z_k = 1 \\ \epsilon^{-1} & \text{for} \quad \zeta_k \in I_\epsilon \;\wedge\; z_k = 0 \\ 0 & \text{otherwise} \end{cases} \\ p(\zeta|\mathbf{z}) &:= \prod_{k=1}^{K} p(\zeta_k|z_k) \\ p(\zeta, \mathbf{z}) &:= p(\zeta|\mathbf{z})p(\mathbf{z}), \end{aligned}$$

where $I_\epsilon := [-2\epsilon, -\epsilon]$ is the refractory resting state interval. In accordance with this definition we can also write $p(\zeta_k|z_k) = z_k \chi_{[0,1]}(\zeta_k) + (1 - z_k)\epsilon^{-1}\chi_{I_\epsilon}(\zeta_k)$.

Lemma 6. *The distribution $p(\zeta, \mathbf{z})$ has the following marginal distribution:*

$$\begin{aligned} p(\zeta_k|\zeta_{\setminus k}) &= \sigma(u_k)\chi_{[0,1]}(\zeta_k) + (1 - \sigma(u_k))\epsilon^{-1}\chi_{I_\epsilon}(\zeta_k) \\ &= \begin{cases} \sigma(u_k) & \text{for} \quad 1 \geq \zeta_k > 0 \\ (1 - \sigma(u_k))\epsilon^{-1} & \text{for} \quad \zeta_k \in I_\epsilon \end{cases}, \end{aligned}$$

where $u_k := u_k(\zeta_{\setminus k}^{>0})$.

Definition 3. *For $k \in \{1, \ldots, K\}$ and $x \in \mathbb{R}$ the operator T_x^k is defined in the following way for a function $q : \mathbb{R} \to \mathbb{R}$:*

$$\begin{aligned} (T_x^k q)(\zeta_k) := \tau^{-1}\Big(&\partial_{\zeta_k}(q(\zeta_k)\chi_{\mathbb{R}^+}(\zeta_k)) - \delta(\zeta_k)F(q) + \exp(x)\delta(\zeta_k - 1)\int_{I_\epsilon} q(\zeta_k')d\zeta_k' \\ &+ \chi_{I_\epsilon}(\zeta_k)\left(\epsilon^{-1}F(q) - \exp(x)q(\zeta_k)\right)\Big). \end{aligned}$$

where the functional F is defined as the one-sided limit from above at 0:

$$F(q) := \lim_{x \to 0^+} q(x).$$

The operator T is defined in the following way for a probability distribution $q(\zeta)$ on $\mathbb{R}^K$:

$$(Tq)(\zeta) := \sum_{k=1}^{K} (T_{u_k}^k q(\zeta_1, \ldots, \zeta_{k-1}, \cdot, \zeta_{k+1}, \zeta_K))(\zeta_k),$$

where $q(\zeta_1, \ldots, \zeta_{k-1}, \cdot, \zeta_{k+1}, \zeta_K) : \mathbb{R} \to \mathbb{R}$ *denotes the function* $q(\zeta)$ *of* ζ_k *where* $\zeta_{\setminus k}$ *is held constant and* $u_k := u_k(\zeta_{\setminus k}^{>0})$.

The transition operator T defines the following Fokker-Planck equation for a time-dependent distribution $q_t(\zeta)$:

$$\partial_t q_t(\zeta) = (T q_t)(\zeta).$$

The jump and drift functions $W^k(\zeta|\zeta')$ and $A^k(\zeta)$ associated to the operator T are given by:

$$\begin{aligned} W^k(\zeta|\zeta') &= \left((\epsilon\tau)^{-1}\chi_{I_\epsilon}(\zeta_k)\delta(\zeta_k') + \delta(\zeta_k - 1)\exp(u_k(\zeta'_{\setminus k}) - \log\tau)\chi_{I_\epsilon}(\zeta_k')\right)\delta(\zeta_{\setminus k} - \zeta'_{\setminus k}) \\ A^k(\zeta) &= -\tau^{-1}\chi_{\mathbb{R}^+}(\zeta_k) \\ \implies \quad (T q_t)(\zeta) &= -\sum_{k=1}^K \partial_{\zeta_k}(A^k(\zeta) q_t(\zeta)) + \sum_{k=1}^K \int \left(W^k(\zeta|\zeta')p(\zeta') - W^k(\zeta'|\zeta)p(\zeta)\right) d\zeta'. \end{aligned}$$

Lemma 7. *The operator* $T_{u_k}^k$ *leaves the conditional distribution* $p(\zeta_k|\zeta_{\setminus k})$ *invariant with* $u_k = u_k(\zeta_{\setminus k}^{>0})$, *i.e.:*

$$(T_{u_k}^k p(\cdot|\zeta_{\setminus k}))(\zeta_k) = 0.$$

Proof. This is easy to proof using calculus and the relations $\partial_{\zeta_k}\chi_{\mathbb{R}^+}(\zeta_k) = \delta(\zeta_k)$ and $F(p(\cdot|\zeta_{\setminus k})) = \sigma(u_k) = \exp(u_k)(1 - \sigma(u_k))$. □

Lemma 8. $p(\zeta)$ *is an invariant distribution of* T, *i.e., it is a solution to the invariant Fokker-Planck equation:*

$$\partial_t p(\zeta) = (T p)(\zeta) = 0.$$

Proof. We observe that $T^k(\alpha p) = \alpha T^k p$ for a constant $\alpha \in \mathbb{R}$ (which is not a function of ζ_k). Hence:

$$\begin{aligned} T_{u_k}^k p(\zeta_1, \ldots, \zeta_{k-1}, \cdot, \zeta_{k+1}, \ldots, \zeta_K) &= T_{u_k}^k(p(\cdot|\zeta_{\setminus k})p(\zeta_{\setminus k})) \\ &= p(\zeta_{\setminus k})(T_{u_k}^k p(\cdot|\zeta_{\setminus k})) \\ &= 0. \end{aligned}$$

The Lemma follows then from the definition of $T := \sum_k T_{u_k}^k$. □

5.5.1.5 Details to neural sampling with a relative refractory period in continuous time

As already assumed in the case of the absolute refractory sampler in continuous time, we define the state space of ζ_k to be $\mathbb{R}^+ \cup [-2\epsilon, -\epsilon]$ for $\epsilon > 0$.

Lemma 9. *Let g be a continuous, non-negative function $g : [0,1] \to \mathbb{R}_0^+$ with $g(\zeta_k) = 1$ for $\zeta_k \leq 0$. There exists a unique $\mathcal{C}^\infty$ function $f : \mathbb{R} \to \mathbb{R}^+$ with the following property $\forall u \in \mathbb{R}$:*

$$f(u) \int_0^1 \exp\left(f(u) \int_0^{\zeta_k} g(\zeta_k') d\zeta_k'\right) d\zeta_k \quad = \quad \exp(u). \tag{5.16}$$

Proof. We define the function $F : \mathbb{R}_0^+ \to \mathbb{R}$ in the following way:

$$F(x) \quad := \quad x \int_0^1 \exp\left(x\alpha(\zeta_k)\right) d\zeta_k,$$

where $\alpha(r) := \int_0^r g(\zeta_k') d\zeta_k'$. From $g(\zeta_k) \geq 0$ we can follow that $\alpha : [0,1] \to \mathbb{R}_0^+$ is non-negative. $F(x)$ is differentiable with the derivative:

$$\begin{aligned} F'(x) \quad &= \quad \int_0^1 \exp\left(x\alpha(\zeta_k)\right) d\zeta_k + x \int_0^1 \exp\left(x\alpha(\zeta_k)\right) \alpha(\zeta_k) d\zeta_k \\ \Rightarrow F'(x) \quad &> \quad 0. \end{aligned}$$

Hence F is strictly monotonously increasing. Furthermore, the following relations hold:

$$\begin{aligned} F(0) \quad &= \quad 0 \\ F(x) \quad &\geq \quad x. \end{aligned}$$

Therefore the equation:

$$F(x) \quad = \quad \exp(u),$$

has exactly one solution $f(u)$ with $F(f(u)) = \exp(u)$ in $\mathbb{R}^+$. From applying the implicit function theorem to $F(x,u) := F(x) - \exp(u)$ it follows that f is $\mathcal{C}^\infty$. □

Definition 4. *For all $k \in \{1, \ldots, K\}$ and $x \in \mathbb{R}$ the operator T_x^k is defined in the following way for a function $q : \mathbb{R} \to \mathbb{R}$:*

$$\begin{aligned} (T_x^k q)(\zeta_k) \quad := \quad \tau^{-1}\Big(&\partial_{\zeta_k}(q(\zeta_k)\chi_{\mathbb{R}^+}(\zeta_k)) - \delta(\zeta_k)q(\zeta_k) + f(x)\delta(\zeta_k - 1) \int_{\mathbb{R}} g(\zeta_k')q(\zeta_k')d\zeta_k' \\ &+ \chi_{I_\epsilon}(\zeta_k)\epsilon^{-1}F(q) - f(x)q(\zeta_k)g(\zeta_k)\Big). \end{aligned}$$

The transition operator T_x^k defines the following Fokker-Planck equation for a time-dependent distribution $q_t(\zeta_k)$:

$$\partial_t q_t(\zeta_k) \quad = \quad (T_x^k q_t)(\zeta_k).$$

The jump and drift functions $W^k(\zeta_k|\zeta_k')$ and $A^k(\zeta_k)$ associated to the operator T_x^k are given by:

$$\begin{aligned} W^k(\zeta_k|\zeta_k') &= (\epsilon\tau)^{-1}\chi_{I_\epsilon}(\zeta_k)\delta(\zeta_k') + \tau^{-1}\delta(\zeta_k - 1)f(x)g(\zeta_k') \\ A^k(\zeta_k) &= -\tau^{-1}\chi_{\mathbb{R}^+}(\zeta_k) \\ \Longrightarrow \quad (T_x^k q_t)(\zeta_k) &= -\partial_{\zeta_k}(A^k(\zeta_k)q_t(\zeta_k)) + \int \left(W^k(\zeta_k|\zeta_k')p(\zeta_k') - W^k(\zeta_k'|\zeta_k)p(\zeta_k)\right) d\zeta_k'. \end{aligned}$$

Lemma 10. *For all $k = 1, \ldots, K$ the invariant distribution $q^*(\zeta_k|\mathbf{z}_{\setminus k})$ of the operator $T_{u_k}^k$ fulfills $\int \delta(z_k, \zeta_k^{>0}) q^*(\zeta_k|\mathbf{z}_{\setminus k}) d\zeta_k = p(z_k|\mathbf{z}_{\setminus k})$.*

Proof. We define the distribution $q^*(\zeta_k|\mathbf{z}_{\setminus k})$ as:

$$q^*(\zeta_k|\mathbf{z}_{\setminus k}) \quad = \quad (1 - \sigma(u_k)) \left(f(u_k)\chi_{[0,1]}(\zeta_k)\exp(f(u_k)\alpha(\zeta_k)) + \epsilon^{-1}\chi_{I_\epsilon}(\zeta_k)\right),$$

where $\alpha(\zeta_k) := \int_0^1 g(\zeta_k')d\zeta_k'$. By applying the operator $T_{u_k}^k$ to q^* one can verify that $T_{u_k}^k q^* = 0$ holds using the definition of $f(u_k)$ given in (5.16). Furthermore we can compute the ratio:

$$\begin{aligned} \frac{\int_0^1 q^*(\zeta_k|\mathbf{z}_{\setminus k})d\zeta_k}{\int_{I_\epsilon} q^*(\zeta_k|\mathbf{z}_{\setminus k})d\zeta_k} &= \frac{p(z_k = 1|\mathbf{z}_{\setminus k})}{p(z_k = 0|\mathbf{z}_{\setminus k})} \\ &= f(u_k)\int_0^1 \exp\left(f(u_k)\int_0^{\zeta_k} g(\zeta_k')d\zeta_k'\right) d\zeta_k = \exp(u_k). \end{aligned}$$

□

5.5.2 Details to the computer simulations

The simulation results shown in Figure 5.2, Figure 5.3 and Figure 5.4 used the biologically more realistic neuron model with the relative refractory mechanism. During all experiments the first second of simulated time was discarded as burn-in time. The full list of parameters defining the experimental setup is given in Table 5.1. All occurring joint probability distributions are Boltzmann distributions of the form given

in equation (5.5). Example Python [220] scripts for neural sampling from Boltzmann distributions are available on request and will be provided on our webpage. The example code comprises networks with both absolute and relative refractory mechanism. It requires standard Python packages only and is readily executable.

5.5.2.1 Details to Figure 2: Neuron model with relative refractory mechanism

The three refractory functions $g(\zeta)$ of panel (B) as well as all other simulation parameters are listed in Table 5.1. Panel (C) shows the corresponding functions $f(u)$, which result from numerically solving equation (5.11). The spike patterns in panel (D) show the response of the neurons when the membrane potential is low ($u_k = -1$ for $0 < t < 250\,\text{ms}$) or high ($u_k = +2$ for $250\,\text{ms} < t < 500\,\text{ms}$). These membrane potentials encode $p(z_k = 1) = 0.269$ and $p(z_k = 1) = 0.881$, respectively according to (5.3) and (5.4). The binary state $z_k = 1$ is indicated by gray shaded areas of duration $\tau \cdot dt = 20\,\text{ms}$ after each spike.

5.5.2.2 Details to Figure 3: Sampling from a Boltzmann distribution by spiking neurons with relative refractory mechanism

We examined the spike response of a network of 40 randomly connected neurons which sampled from a Boltzmann distribution. The excitabilities b_k as well as the the synaptic weights $W_{ki} (= W_{ik})$ were drawn from Gaussian distributions (with diagonal elements $W_{ii} = 0$). For the full list of parameters please refer to Table 5.1. One second of the arising spike pattern is shown in panel (A). The average firing rate of the network was 13.9 Hz. To highlight the internal dynamics of the neuron model, the values of the refractory function $g(\zeta_{26})$, the membrane potential u_{26} and the instantaneous firing rate r_{26} of neuron ν_{26} (indicated with red spikes) are shown in panel (B). Here, the instantaneous firing rate r_{26} is defined for the discrete time Markov chain as

$$r_{26} = p(\text{spike})/dt = T^{26}(\tau|\zeta_{26}, \mathbf{z}_{\setminus 26})/dt = g(\zeta_{26}) \cdot f(u_{26})/dt \ . \tag{5.17}$$

As stated before, the neuron model with relative refractory mechanism $g_k(\zeta)$ does not entail the correct overall invariant distribution $p(\mathbf{z})$. To estimate the impact of this approximation on the joint network dynamics, we compared the distribution $p(z_{24}, \ldots, z_{28})$ over five neurons (indicated by gray background in A) in the spiking

network with the correct distribution obtained from Gibbs sampling. The probabilities were estimated from 10^7 samples. A more quantitative analysis of the approximation quality of neural sampling with a relative refractory mechanism is provided below.

5.5.2.3 Details to Figure 4: Modeling perceptual multistability as probabilistic inference with neural sampling

We demonstrate probabilistic inference and learning in a network of orientation selective neurons. As a simple model we consider a network of 217 neurons on a hexagonal grid as shown in panel (F). Any two neurons with distance ≤ 8 were synaptically connected (neighboring units had distance 1). For the remaining parameters of the network and neuron model please refer to Table 5.1. Each neuron featured a π-periodic tuning curve as depicted in panel (B):

$$V_k(\varphi) = v_0 + C \cdot \exp\left[\kappa \cdot \cos\left(2(\varphi - \bar{\varphi}_k)\right) - \kappa\right] \tag{5.18}$$

with base sensitivity v_0, contrast C, peakedness κ and preferred orientation $\bar{\varphi}_k$. The preferred orientations $\bar{\varphi}_k$ of the neurons were chosen to cover the entire interval $[0, \pi)$ of possible orientations with equal spacing and were randomly assigned to the neurons.

For simplicity we did not incorporate the input dynamics in our probabilistic model, but rather trained the network directly like a fully visible Boltzmann machine. We used for this purpose a standard Boltzmann machine learning rule known as contrastive divergence [97, 98]. This learning rule requires posterior samples $\tilde{\mathbf{z}}$, i.e., network states under the influence of the present input, and approximate prior samples $\mathbf{z}^\star$, which reflect the probability distribution of the network in the absence of stimuli. The update rules for synaptic weights and neuronal excitabilities read:

$$\begin{aligned} \Delta W_{ki} &= \eta_{ki} \cdot (\tilde{z}_k \tilde{z}_i - z_k^\star z_i^\star) \\ \Delta b_k &= \eta \cdot (\tilde{z}_k - z_k^\star) \\ \eta_{ki} &= \begin{cases} \eta & \text{if } \nu_k \text{ and } \nu_i \text{ are connected} \\ 0 & \text{otherwise} \end{cases}. \end{aligned} \tag{5.19}$$

While more elaborate policies can speed up convergence, we simply used a global learning rate η which was constant in time. The values of W_{ki} and b_k were initialized at 0. We generated binary training patterns in the following way:

1. A global orientation φ was drawn uniformly from $[0, \pi)$,

2. each neuron was independently set to be active with probability $p(z_k = 1) = V_k(\varphi)$,

3. the resulting network state $\tilde{\mathbf{z}}$ was taken as posterior sample.

To obtain an approximate prior sample $\mathbf{z}^\star$ we let the network run for a short time freely starting from $(\tilde{\zeta}, \tilde{\mathbf{z}})$. The variables $\tilde{\zeta}$ were also assumed to be observed with $\tilde{\zeta}_k \sim$ iid. uniformly in $\{1, \ldots, \tau\}$ if $\tilde{z}_k = 1$ and $\tilde{\zeta}_k = 0$ otherwise. After evolving freely for 20 time steps, the resulting network state $\mathbf{z}^\star$ was taken as approximate prior sample and $\mathbf{W}$ and $\mathbf{b}$ were updated according to (5.19). This process was repeated $N_{\text{train}} = 10^5$ times. As a result, neurons with similar preferred orientations featured excitatory synaptic connections ($W_{ki} = 6.4 \cdot 10^{-3} \pm 6.7 \cdot 10^{-3}$ = mean $\pm$ standard deviation of weight distribution), those with dissimilar orientations maintained inhibitory synapses ($W_{ki} = -4.9 \cdot 10^{-3} \pm 5.2 \cdot 10^{-3}$). Here, preferred orientations $\bar{\varphi}_i$ and $\bar{\varphi}_j$ are defined as similar if $V_i(\bar{\varphi}_j) - v_0 = V_j(\bar{\varphi}_i) - v_0 > 0.5\,C$, otherwise they are dissimilar. Neuronal biases converged to $b_k = -0.08 \pm 0.03$.

We illustrate the learned prior distribution $p(\mathbf{z})$ of the network through sampled states when the network evolved freely. As seen in panel (D), the population vector – a 2-dimensional projection of the high dimensional network state – typically reflected an arbitrary, yet coherent, orientation (for the definition of the population vector see below). Each dot represents a sampled network state $\mathbf{z}$.

To apply an ambiguous cue, we clamped 8 out of 217 neurons: Two units with $\bar{\varphi}_k \approx \pi/4$ and two with $\bar{\varphi}_k \approx 3\pi/4$ were set active, two units with $\bar{\varphi}_k \approx 0$ and two with $\bar{\varphi}_k \approx \pi/2$ were set inactive. This led to a bimodal posterior distribution as shown in panel (E). The sampling network represented this distribution by encoding either global perception separately: The trace of network states $\mathbf{z}(t)$ roamed in one mode for multiple steps before quickly crossing the state space towards the opposite percept.

We define the population vector $\mathbf{x}$ of a network state $\mathbf{z}$ as a function of the preferred orientations of all active units:

$$\mathbf{x} = (x_0, x_{\pi/4}) = \sum_{k=1}^{K} z_k \cdot (\cos 2\tilde{\varphi}_k, \sin 2\tilde{\varphi}_k) \quad . \tag{5.20}$$

This definition of $\mathbf{x}$ is not based on the preferred orientations $\bar{\varphi}_k$ which are used for generating external input to the network from a given stimulus with orientation φ. It

is rather based on the preferred orientations $\tilde{\varphi}_k$ measured from the network response. We used population vector decoding based on the measured values $\tilde{\varphi}_k$, as they are conceptually closer to experimentally measurable preferred orientations, and this decoding hence does not require knowledge of the (unobservable) $\bar{\varphi}_k$. For every neuron ν_k the preferred orientation $\tilde{\varphi}_k$ was measured in the following way. We estimated a tuning curve $\tilde{V}_k(\varphi)$ by a van-Mises fit (of the form (5.18)) to data from stimulation trials in which neuron ν_k was not clamped, i.e., where ν_k was only stimulated by recurrent input (feedforward input was modeled by clamping 8 out of 217 neurons as a function of stimulus orientation φ as before). Due to the structured recurrent weights, the experimentally measured tuning curves $\tilde{V}_k(\varphi)$ were found to be reasonably close to the tuning curves $V_k(\varphi)$ used for external stimulation. $\tilde{\varphi}_k$ was set to the preferred orientation of $\tilde{V}_k(\varphi)$ (localization parameter of the van-Mises fit). The measured values $\tilde{\varphi}_k$ turned out to be consistent with the preferred orientations $\bar{\varphi}_k$ ($\bar{\varphi}_k - \tilde{\varphi}_k = 6 \cdot 10^{-4} \pm 8.3 \cdot 10^{-3}$ averaged over all K neurons). The mean and standard deviation of the remaining parameter values v_0, C and κ of the fitted tuning curves $\tilde{V}_k(\varphi)$ are listed in Table 5.1 next to the ones used for stimulation.

The population vector $\mathbf{x}$ was defined in (5.20) with the argument $2\tilde{\varphi}_k$ (instead of $\tilde{\varphi}_k$) as orthogonal orientations should cancel each other and neighborhood relations should be respected. For example neurons with $\tilde{\varphi}_k = \epsilon$ and $\tilde{\varphi}_k = \pi - \epsilon$ contribute similarly to the population vector for small ϵ. But counter to intuition the population vector of a state $\mathbf{z}$ with dominant orientation φ_z will point into direction $\varphi_x = 2\varphi_z$. For visualization in panel (D) and (E) we therefore rescaled the population vector: If $(x_0, x_{\pi/4}) \mapsto (r_x, \varphi_x)$ in polar coordinates, then the dot is located at $(r_x, \varphi_x/2)$ in accord with intuition. The black semicircles equal $|\mathbf{x}| = r_x = 45$.

The population vector $(x_0, x_{\pi/4}) \in \mathbb{R}^2$ was also used for measuring the dominance durations shown in panel (C). To this $\mathbb{R}^2$ was divided into 3 areas: (a) $x_{\pi/4} < -35$, (b) $-35 \leq x_{\pi/4} \leq 35$, (c) $35 < x_{\pi/4}$. We detected a perceptual switch when the network state entered area (a) or (c) while the previous perception was (c) or (a), respectively.

In panel (F) neurons ν_k with $z_k = 1$ are plotted with their preferred orientation color code, inactive neurons are displayed in white. Cells marked by a dot (•) were part of the observed variables $\mathbf{o}$. The three network states correspond to $\mathbf{z}(t_i)$ with $t_1 = 100\,\text{ms}$, $t_2 = 250\,\text{ms}$ and $t_3 = 400\,\text{ms}$ in the spike pattern in panel (G). The spike pattern shows the response of the freely evolving units around a perceptual switch during sampling

from the posterior distribution. The corresponding trace of the population vector is drawn as black line in panel (E). The width of the light-gray shaded areas in the spike pattern equals the PSP duration $\tau \cdot dt$, i.e., neurons that spiked in these intervals were active in the corresponding state in (F).

5.5.3 Firing statistics of neural sampling networks

In previous sections it was shown that a spiking neural network can draw samples from a given joint distribution which is in a well-defined class of probability distributions (see the neural computability condition (5.4)). Here, we examine some statistics of individual neurons in a sampling network which are commonly used to analyze experimental data from recordings. The spike trains and membrane potential data are taken from the simulation presented in Figure 5.3.

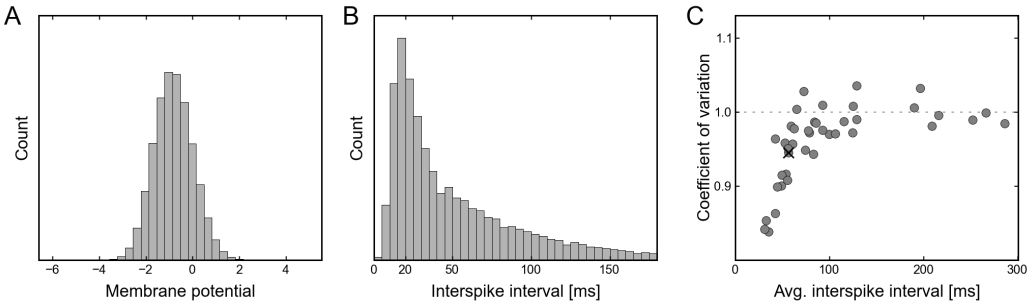

Figure 5.5: Firing statistics of neural sampling networks. (A) Shown is the membrane potential histogram of a typical neuron during sampling. The data is that of neuron ν_{26} from the simulation shown in Figure 5.3 (the membrane potential and spike trace of ν_{26} are highlighted in Figure 5.3). (B) The plot shows the ISI distribution of a typical neuron (again ν_{26} from Figure 5.3) during sampling. The distribution is roughly gamma-shaped, reminiscent of experimentally observed ISI distributions. (C) A scatter plot of the coefficient of variation (CV) versus the average interspike interval (ISI) of each neuron taken from the simulation shown in Figure 5.3. The value of neuron ν_{26} from Figure 5.3 is marked by a cross. The simulated data is in accordance with experimentally observed data.

Figure 5.5A,B exemplarily show the distribution of the membrane potential u_k and the interspike interval (ISI) histogram of a single neuron, namely neuron ν_{26} which was already considered in Figure 5.3B. The responses of other neurons yield qualitatively similar statistics. The bell-shaped distribution of the membrane potential is commonly

observed in neurons embedded in an active network [178]. The ISI histogram reflects the reduced spiking probability immediately after an action potential due the refractory mechanism. Interspike intervals larger than the refractory time constant $\tau \cdot dt = 20\,\text{ms}$ roughly follow an exponential distribution. Similar ISI distributions were observed during in-vivo recordings in awake, behaving monkeys [205].

Figure 5.5C shows a scatterplot of the coefficient of variation (CV) of the ISIs versus the average ISI for each neuron in the network. The neurons exhibited a variety of average firing rates between 3.5 Hz and 31.5 Hz. Most of the neurons responded in a highly irregular manner with a CV ≈ 1. Neurons with high firing rates had a slightly lower CV due to the increased influence of the refractory mechanism The dashed line marks the CV of a Poisson process, i.e., a memoryless spiking behavior. The CV of neuron ν_{26} is marked by a cross. The structure of this plot resembles, e.g., data from recordings in behaving macaque monkeys [208] (but note the lower average firing rate).

5.5.4 Approximation quality of neural sampling with different neuron and synapse models

The theory of the neuron model with absolute refractory mechanism guarantees sampling form the correct distribution. In contrast, the theory for the neuron model with a relative refractory mechanism only shows that the sampling process is "locally correct", i.e., that it would yield correct conditional distributions $p(z_k|\mathbf{z}_{\setminus k})$ for each individual neuron if the state of the remaining network $\mathbf{z}_{\setminus k}$ stayed constant. Therefore, the stationary distribution of the sampling process with relative refractory mechanism only provides an approximation to the target distribution. In the following we examine the approximation quality and robustness of sampling networks with different refractory mechanisms for target Boltzmann distributions with parameters randomly drawn from different distributions. Furthermore, we investigate the effect of additive PSP shapes with more realistic time courses.

We generated target Boltzmann distributions with randomly drawn weights W_{ki} and biases (excitabilities) b_k and computed the similarity between these reference distributions and the corresponding neural sampling approximations. The setup of these simulations is the same as for the simulation presented in Figure 5.3. As we aimed to compare the distribution $q^*(\mathbf{z})$ sampled by the network with the exact Boltzmann distribution $p(\mathbf{z})$, we reduced the number of neurons per network to $K = 10$. This

resulted in a state space of 2^{10} possible network states $\mathbf{z}$ for which the normalization constant for the target Boltzmann distribution could be computed exactly. The weight matrix W was constraint to be symmetric with vanishing diagonal. Off-diagonal elements were drawn from zero-mean normal distributions with three different standard deviations $\sigma = 0.03$, $\sigma = 0.3$ and $\sigma = 3$, whereas the b_k were sampled from the same distribution as in Figure 5.3. For every value of the hyperparameter σ we generated 100 random distributions. For Boltzmann distributions with small weights ($\sigma = 0.03$), the RVs are nearly independent, whereas distributions with intermediate weights ($\sigma = 0.3$) show substantial statistical dependencies between RVs. For very large weights ($\sigma = 3$), the probability mass of the distributions is concentrated on very few states (usually 90% on less than 10 out of the 2^{10} states). Hence, the range of the hyperparameter $0.03 \leq \sigma \leq 3$ considered here covers a range a very different distributions.

The approximation quality of the sampled distribution was measured in terms of the Kullback-Leibler divergence between the target distribution p and the neural approximation q^*

$$D_{\mathrm{KL}}(p||q^*) \quad = \quad \sum_{\mathbf{z}} p(\mathbf{z}) \log \frac{p(\mathbf{z})}{q^*(\mathbf{z})} \quad . \tag{5.21}$$

We estimated q^* from 10^7 samples for each simulation trial using a Laplace estimator, i.e., we added a priori 1 to the number of occurrences of each state $\mathbf{z}$.

Table 5.2 shows the means and the standard deviations of the Kullback-Leibler divergences between the target Boltzmann distributions and the estimated approximations stemming from neural sampling networks with three different neuron and synapse models: the exact model with absolute refractory mechanism and two models with different relative refractory mechanisms shown in the bottom and middle row in Figure 5.2B. Additionally, as a reference, we provide the (analytically calculated) Kullback-Leibler divergences for fully factorized distributions, i.e., $q^*(\mathbf{z}) = \prod_k q^*(z_k)$ with correct marginals $q^*(z_k) = p(z_k)$ but independent variables $z_i,\ z_j$ for $i \neq j$.

The absolute refractory model provides the best results as we expected due to the theoretical guarantee to sample from the correct distribution (the non-zero Kullback-Leibler divergence is caused by the estimation from a finite number of samples). The models with relative refractory mechanism provide faithful approximations for all values of the hyperparameter σ considered here. These relative refractory models are characterized by the theory to be "locally correct" and turn out to be much more accurate

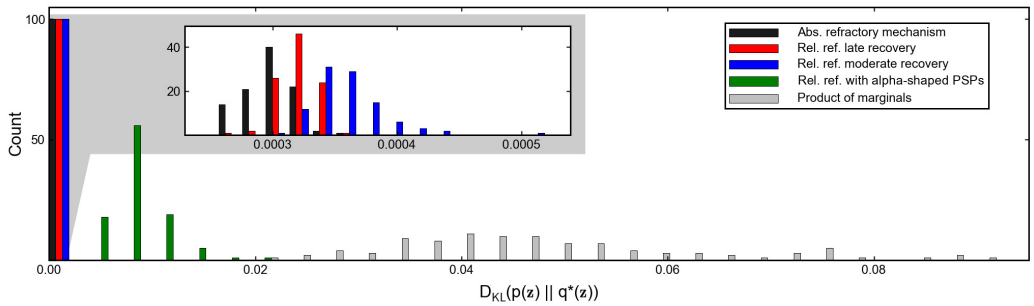

Figure 5.6: Comparison of neural sampling with different neuron and synapse models. The figure shows a histogram of the Kullback-Leibler divergence between 100 different Boltzmann distributions over K = 10 variables (with parameters randomly drawn, see setup of Figure 5.3) and approximations stemming from different neural sampling networks. Networks with absolute refractory mechanism provide the best approximation (as expected from theoretical guarantees). Networks consisting of neurons with relative refractory mechanisms, with only "locally" correct sampling, also provide a close fit to the true distribution (see inset) compared to a fully factorized approximation (assuming correct marginals and independent variables). Furthermore, it can be seen that sampling networks with more realistic, alpha-shaped, additive PSPs still fit the true distribution reasonably well.

approximations than fully factorized distributions if substantial statistical dependencies between the RVs are present (i.e., $\sigma = 0.3$, $\sigma = 3$). As expected, a late recovery of the refractory function $g(\zeta)$ is beneficial for the approximation quality of the model as it is closer to an absolute refractory mechanism. Figure 5.6 explicitly shows the full histograms of the Kullback-Leibler divergences for the intermediate weights group ($\sigma = 0.3$). Systematic deviations due to the relative refractory mechanism are on the same order as the effect of estimating from finite samples (as can be seen, e.g., from a comparison with the absolute refractory model which has 0 systematic error). For completeness, we mention that the divergences of the fully factorized distributions of 2 out of the 100 networks with $D_{\mathrm{KL}} > 0.1$ are not shown in the plot.

The theorems presented in this article assumed renewed (i.e., non-additive), rectangular PSPs. In the following we examine the effect of additive PSPs with more realistic time courses. We define additive, alpha-shaped PSPs in the following way. The influence Δu_{ki} of each presynaptic neuron ν_i on the postsynaptic membrane potential u_k is

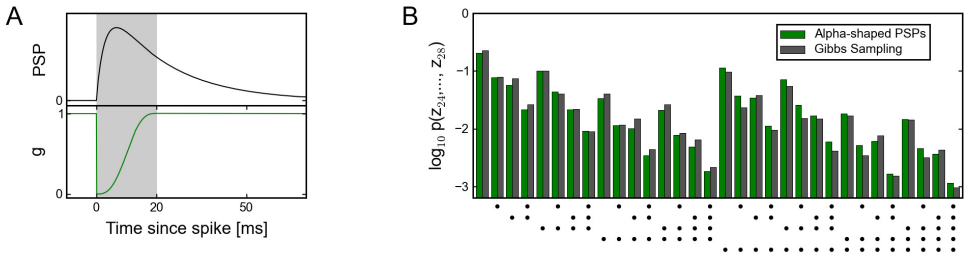

Figure 5.7: Sampling from a Boltzmann distribution with more realistic PSP shapes. (A) The upper panel shows the shape of a single PSP elicited at time $t = 0$. The lower panel shows the time course of the refractory function $g(\zeta_k(t))$ caused by a single spike of neuron ν_k at $t = 0$. The grey-shaded area of length $\tau \cdot dt = 20\,ms$ indicates the interval of neuron ν_k being active (i.e., $z_k = 1$) due to a single spike of neuron ν_k at time $t = 0$. (B) Shown is the probability distribution of 5 out of 40 neurons. The plot is similar to Figure 5.3C, however it is generated with a sampling network that features alpha-shaped, additive PSPs. It can be seen that the network still produces a reasonable approximation to the true Boltzmann distribution (determined by Gibbs sampling).

modeled by convolving the input spikes with a kernel κ:

$$\Delta u_{ki}(t) \quad = \quad W_{ki} \cdot \sum_f \kappa(t - t_i^f) \tag{5.22}$$

where $\kappa(s) = \lambda \cdot (e^{-s/\tau_+} - e^{-s/\tau_-})$ for $s \geq 0$ and $\kappa(s) = 0$ for $s < 0$, and t_i^f for $f \in \mathbb{N}$ are the spike times of the presynaptic neuron ν_i. The time constant governing the rising edge of the PSPs was set to $\tau_- = 3\,\text{ms}$. The time constant controlling the falling edge was chosen equal to the duration of rectangular PSPs, $\tau_+ = \tau \cdot dt = 20\,\text{ms}$. The scaling parameter λ was set such that the time integral over a single PSP matches the time integral over the theoretically optimal rectangular PSP, i.e., $\lambda = \tau \cdot dt/(\tau_+ - \tau_-) = 20/17$. These parameters display a simple and reasonable choice for the purpose of this study (an optimization of λ, τ_+ and τ_- is likely to yield an improved approximation quality). Figure 5.7A shows the resulting shape of the non-rectangular PSP. Furthermore the time course of the function $g(\zeta_k(t))$ caused by a single spike of neuron ν_k is shown in order to illustrate that the time constants of g and of a PSP are closely related due to the assumption $\tau_+ = \tau \cdot dt$ made above. Preliminary and non-exhaustive simulations seem to suggest that the choice $\tau_+ = \tau \cdot dt$ yields better approximation quality than setting $\tau_+ \gg \tau \cdot dt$ or $\tau_+ \ll \tau \cdot dt$; however it is very well possible that a mismatch

between τ_+ and $\tau \cdot dt$ can be compensated for by adapting other parameters, e.g., the PSP magnitude or a specific choice of the refractory function g. Figure 5.7B shows the results of an experiment, similar to the one presented in Figure 5.3C, with additive, alpha-shaped PSPs and relative refractory mechanism. While differences to Gibbs sampling results are visible, the spiking network still captures dependencies between the binary random variables quite well.

For a quantitative analysis of the approximation quality, we repeated the experiment of Figure 5.6 with additive, alpha-shaped PSPs (shown as green bars). The Kullback-Leibler divergence $D_{\mathrm{KL}}(p||q^*)$ to the true distribution is clearly higher compared to the case of renewed, rectangular PSPs. Still networks with this more realistic synapse model account for dependencies between the random variables $\mathbf{z}$ and yield a better approximation of $p(\mathbf{z})$ than fully factorized distributions.

5.6 Tables

Table 5.1: List of parameters of the computer simulations

Description	Variable	Value	Figure	Comment
Simulation Time				
Simulation step size	dt	1 ms	2-7	interpretation of an MCMC step
Burn-in time	t_{burn}	1 s	2-7	before recording spikes
Simulation time	t_{sim}	0.5 s	2	
		10^4 s	3,5-7	
		20 s	4	10^4 s for Figure 5.4C
Network				
Number of neurons	K	3	2	unconnected
		40	3,5,6	randomly connected
		217	4	
		10	7	100 networks
Connection radius		0	2	
		∞	3,5-7	
		8	4	
Recurrent weights	W_{ki}	$\mathcal{N}(0,\, 0.3^2)$	3,5-7	from Gaussian distribution
Falling edge	τ_+	20 ms	6,7	for realistic PSP shapes
Rising edge	τ_-	3 ms	6,7	
Scaling factor	λ	20/17	6,7	
Neuron Model				
Number recovery steps	τ	20	2-7	PSP duration $= \tau \cdot dt = 20$ ms
Refractory function	$g(\zeta)$	$\left[4(1-\zeta) + \frac{1}{2\pi}\sin(8\pi\zeta)\right]$	2↑	normalized to $\zeta \in [0, 1]$,
		$\left[1 - \zeta + \frac{1}{2\pi}\sin(2\pi\zeta)\right]$	2-7	$[x] := \min\{1, \max\{0, x\}\}$
		$\left[1 - 2\zeta + \frac{1}{2\pi}\sin(4\pi\zeta)\right]$	2↓,7	
Excitability	b_k	-1 or 2	2	defines membrane potential u_k
		$\mathcal{N}(-1.5,\, 0.5^2)$	3,5-7	from Gaussian distribution
		0	4	initial value
Tuning Function, Training and Inference (Figure 4)				
Peakedness	κ	3	4	measured: 1.78 ± 0.15
Base sensitivity	v_0	0.05	4	measured: 0.017 ± 0.009
Sensitivity contrast	C	0.9	4	measured: 0.760 ± 0.020
Training samples	N_{train}	10^5	4	
Decorrelation steps		20	4	for contrastive divergence
Learning rate	η	10^{-4}	4	
Number of neurons clamped on/off		4/4	4	

Table 5.2: Approximation quality of networks with different refractory mechanisms

σ	**Absolute refract.**	**Rel. late recovery**	**Rel. moderate rec.**	**Prod. of marginals**
0.03	$(3.10 \pm 0.18) \cdot 10^{-4}$	$(3.21 \pm 0.15) \cdot 10^{-4}$	$(3.33 \pm 0.17) \cdot 10^{-4}$	$(4.65 \pm 1.28) \cdot 10^{-4}$
0.3	$(2.98 \pm 0.19) \cdot 10^{-4}$	$(3.20 \pm 0.15) \cdot 10^{-4}$	$(3.58 \pm 0.3) \cdot 10^{-4}$	$(4.94 \pm 1.91) \cdot 10^{-2}$
3.0	$(1.32 \pm 0.45) \cdot 10^{-4}$	$(4.20 \pm 8.70) \cdot 10^{-3}$	$(1.00 \pm 1.82) \cdot 10^{-2}$	$(5.36 \pm 6.71) \cdot 10^{-1}$

Mean and standard deviation of the Kullback-Leibler divergence $D_{\mathrm{KL}}(p||q^*)$ between reference Boltzmann distributions p and neural sampling approximations q^* for three different neuron models (corresponding to columns) and three different values for the reference distribution hyperparameter σ (corresponding to rows). The parameter σ controls the standard deviation of the weights of the reference distributions $p(\mathbf{z})$. In case of very strong synaptic interactions (leading to sharply peaked distributions, $\sigma = 3$) the approximation quality of the spiking network degrades, if the neurons feature a relative refractory mechanism. The data was computed from 100 randomly generated Boltzmann distributions and their neural approximations for each value of σ.

6

Homeostatic plasticity in Bayesian spiking networks as Expectation Maximization with posterior constraints

Recent spiking network models of Bayesian inference and unsupervised learning frequently assume either inputs to arrive in a special format or employ complex computations in neuronal activation functions and synaptic plasticity rules. Here we show in a rigorous mathematical treatment how homeostatic processes, which have previously received little attention in this context, can overcome common theoretical limitations and facilitate the neural implementation and performance of existing models. In particular, we show that homeostatic plasticity can be understood as the enforcement of a 'balancing' posterior constraint during probabilistic inference and learning with Expectation Maximization. We link homeostatic dynamics to the theory of variational inference, and show that nontrivial terms, which typically appear during probabilistic inference in a large class of models, drop out. We demonstrate the feasibility of our approach in a spiking Winner-Take-All architecture of Bayesian inference and learning. Finally, we sketch how the mathematical framework can be extended to richer recurrent network architectures. Altogether, our theory provides a

novel perspective on the interplay of homeostatic processes and synaptic plasticity in cortical microcircuits, and points to an essential role of homeostasis during inference and learning in spiking networks.

6.1 Introduction

Experimental findings from neuro- and cognitive sciences have led to the hypothesis that humans create and maintain an internal model of their environment in neuronal circuitry of the brain during learning and development [16, 62, 127, 169], and employ this model for Bayesian inference in everyday cognition [6, 87]. Yet, how these computations are carried out in the brain remains largely unknown. A number of innovative models has been proposed recently which demonstrate that in principle, spiking networks can carry out quite complex probabilistic inference tasks [27, 45, 172, 211], and even learn to adapt to their inputs near optimally through various forms of plasticity [25, 46, 116, 159, 186]. Still, in network models for concurrent online inference and learning, most approaches introduce distinct assumptions: Both [159] in a spiking Winner-take-all (WTA) network, and [116] in a rate based WTA network, identified the limitation that inputs must be normalized before being presented to the network, in order to circumvent an otherwise nontrivial (and arguably non-local) dependency of the intrinsic excitability on all afferent synapses of a neuron. Nessler et al. [159] relied on population coded input spike trains; Keck et al. [116] proposed feed-forward inhibition as a possible neural mechanism to achieve this normalization. A theoretically related issue has been encountered by Deneve [45, 46], in which inference and learning is realized in a two-state Hidden Markov Model by a single spiking neuron. Although synaptic learning rules are found to be locally computable, the learning update for intrinsic excitabilities remains intricate. In a different approach, Brea et al. [25] have recently proposed a promising model for Bayes optimal sequence learning in spiking networks in which a global reward signal, which is computed from the network state and synaptic weights, modulates otherwise purely local learning rules. Also the recent innovative model for variational learning in recurrent spiking networks by Rezende et al. [186] relies on sophisticated updates of variational parameters that complement otherwise local learning rules.

There exists great interest in developing Bayesian spiking models which require minimal non-standard neural mechanisms or additional assumptions on the input distribution: such models are expected to foster the analysis of biological circuits from a Bayesian perspective [219], and to provide a versatile computational framework for novel neuromorphic hardware [201]. With these goals in mind, we introduce here a novel theoretical perspective on homeostatic plasticity in Bayesian spiking networks that complements previous approaches by constraining statistical properties of the *network response* rather than the input distribution. In particular we introduce *'balancing' posterior constraints* which can be implemented in a purely local manner by the spiking network through a simple rule that is strongly reminiscent of homeostatic intrinsic plasticity in cortex [48, 231]. Importantly, it turns out that the emerging network dynamics eliminate a particular class of nontrivial computations that frequently arise in Bayesian spiking networks.

First we develop the mathematical framework for Expectation Maximization (EM) with homeostatic posterior constraints in an instructive Winner-Take-all network model of probabilistic inference and unsupervised learning. Building upon the theoretical results of [83], we establish a rigorous link between homeostatic intrinsic plasticity and variational inference. In a second step, we sketch how the framework can be extended to recurrent spiking networks; by introducing posterior constraints on the correlation structure, we recover local plasticity rules for recurrent synaptic weights.

6.2 Homeostatic plasticity in WTA circuits as EM with posterior constraints

We first introduce, as an illustrative and representative example, a generative mixture model $p(\boldsymbol{z}, \boldsymbol{y}|\boldsymbol{V})$ with hidden causes $\boldsymbol{z}$ and binary observed variables $\boldsymbol{y}$, and a spiking WTA network $\mathcal{N}$ which receives inputs $\boldsymbol{y}(t)$ via synaptic weights $\boldsymbol{V}$. As shown in [159], such a network $\mathcal{N}$ can implement probabilistic inference $p(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{V})$ through its spiking dynamics, and maximum likelihood learning through local synaptic learning rules (see Figure 1A). The mixture model comprises K binary and mutually exclusive components $z_k \in \{0, 1\}$, $\sum_{k=1}^{K} z_k = 1$, each specialized on a different N-dimensional

input pattern:

$$p(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{V}) = \prod_{k=1}^{K} e^{\hat{b}_k z_k} \prod_{i=1}^{N} \left[(\pi_{ki})^{y_i} \cdot (1-\pi_{ki})^{1-y_i}\right]^{z_k} \tag{6.1}$$

$$\Leftrightarrow \log p(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{V}) = \sum_k z_k \left(\sum_i V_{ki} y_i - A_k + \hat{b}_k \right) \ , \tag{6.2}$$

$$\text{with } \sum_k e^{\hat{b}_k} = 1 \text{ and } \pi_{ki} = \sigma(V_{ki}) \text{ and } A_k = \sum_i \log(1 + e^{V_{ki}}) \ , \tag{6.3}$$

where $\sigma(x) = (1 + \exp(-x))^{-1}$ denotes the logistic function, and π_{ki} the expected activation of input i under the mixture component k. For simplicity and notational convenience, we will treat the prior parameters $\hat{b}_k$ as constants throughout the paper. Probabilistic inference of hidden causes z_k based on an observed input $\boldsymbol{y}$ can be implemented by a spiking WTA network $\mathcal{N}$ of K neurons which fire with the instantaneous spiking probability (for $\delta t \to 0$),

$$p(z_k \text{ spikes in } [t, t+\delta t]) = \delta t \cdot r_{\text{net}} \cdot \frac{e^{u_k(t)}}{\sum_j e^{u_j(t)}} \propto p(z_k = 1|\boldsymbol{y}, \boldsymbol{V}) \ , \tag{6.4}$$

with the input potential $u_k(t) = \sum_i V_{ki} y_i(t) - A_k + \hat{b}_k$. Each WTA neuron k receives spiking inputs y_i via synaptic weights V_{ki} and responds with an instantaneous spiking probability which depends exponentially on its input potential u_k in accordance with biological findings [113]. Stochastic winner-take-all (soft-max) competition between the neurons is modeled via divisive normalization (6.4) [206]. The input is defined as $y_i(t) = 1$ if input neuron i emitted a spike within the last τ milliseconds, and 0 otherwise, corresponding to a rectangular post-synaptic potential (PSP) of length τ. We define $z_k(t) = 1$ at spike times t of neuron k and $z_k(t) = 0$ otherwise.

In addition to the spiking input, each neuron's potential u_k features an intrinsic excitability $-A_k + \hat{b}_k$. Note that, besides the prior constant $\hat{b}_k$, this excitability depends on the normalizing term A_k, and hence on all afferent synaptic weights through (6.3): WTA neurons which encode strong patterns with high probabilities π_{ki} require lower intrinsic excitabilities, while neurons with weak patterns require larger excitabilities. In the presence of synaptic plasticity, i.e., time-varying V_{ki}, it is unclear how biologically realistic neurons could communicate ongoing changes in synaptic weights from distal synaptic sites to the soma. This critical issue was apparently identified in [159] and [116]; both papers circumvent the problem (in similar probabilistic models) by

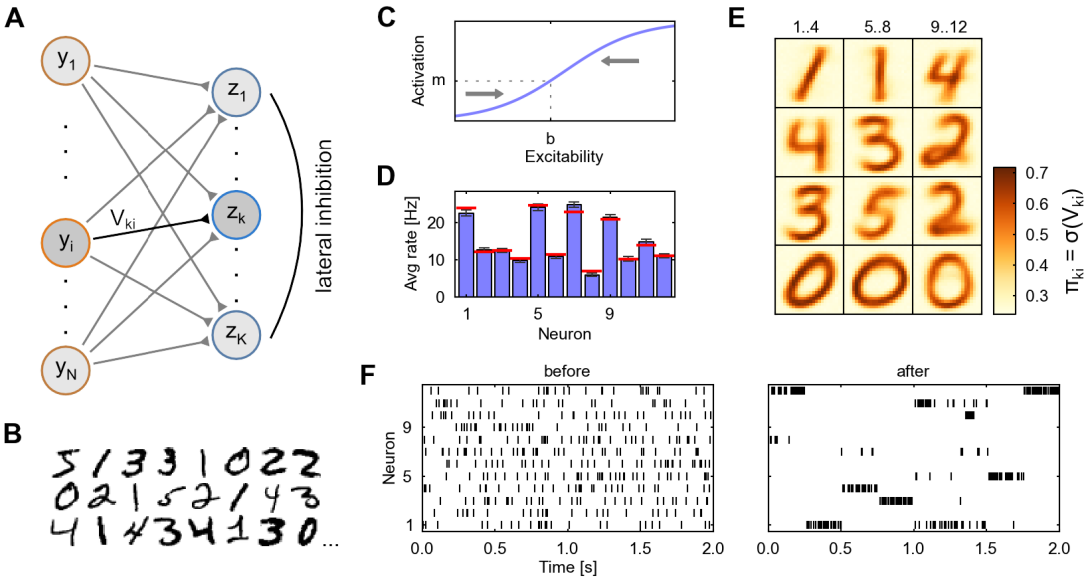

Figure 6.1: **A**. Spiking WTA network model. **B**. Input templates from MNIST database (digits *0-5*) are presented in random order to the network as spike trains (the input template switches after every 250ms, black/white pixels are translated to high/low firing rates between 20 and 90 Hz). **C**. Sketch of intrinsic homeostatic plasticity maintaining a certain target average activation. **D**. Homeostatic plasticity induces average firing rates (blue) close to target values (red). **E**. After a learning period, each WTA neuron has specialized on a particular input motif. **F**. WTA output spikes during a test phase before and after learning. Learning leads to a sparse output code.

constraining the input $\boldsymbol{y}$ (and also the synaptic weights in [116]) in order to maintain constant and uniform values A_k across all WTA neurons.

Here, we propose a different approach to cope with the nontrivial computations A_k during inference and learning in the network. Instead of assuming that the inputs $\boldsymbol{y}$ meet a normalization constraint, we constrain the *network response* during inference, by applying homeostatic dynamics to the intrinsic excitabilities. This approach turns out to be beneficial in the presence of time-varying synaptic weights, i.e., during ongoing changes of V_{ki} and A_k. The resulting interplay of intrinsic and synaptic plasticity can be best understood from the standard EM lower bound [22],

$$F(\boldsymbol{V}, q(\boldsymbol{z}|\boldsymbol{y})) = L(\boldsymbol{V}) - \langle \mathrm{KL}\left(q(\boldsymbol{z}|\boldsymbol{y}) \,||\, p(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{V})\right) \rangle_{p^*(\boldsymbol{y})} \qquad \rightarrow \text{E-step} \;, \tag{6.5}$$

$$= \langle \log p(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{V}) \rangle_{p^*(\boldsymbol{y})q(\boldsymbol{z}|\boldsymbol{y})} + \langle H(q(\boldsymbol{z}|\boldsymbol{y})) \rangle_{p^*(\boldsymbol{y})} \qquad \rightarrow \text{M-step} \;, \tag{6.6}$$

where $L(\boldsymbol{V}) = \langle \log p(\boldsymbol{y}|\boldsymbol{V}) \rangle_{p^*(\boldsymbol{y})}$ denotes the log-likelihood of the input under the model,

$\mathrm{KL}(\cdot\,||\,\cdot)$ the Kullback-Leibler divergence, and $H(\cdot)$ the entropy. The decomposition holds for arbitrary distributions q. In hitherto proposed neural implementations of EM [46, 116, 159, 198], the network implements the current posterior distribution in the E-step, i.e., $q = p$ and $\mathrm{KL}(q\,||\,p) = 0$. In contrast, by applying homeostatic plasticity, the network response will be constrained to implement a variational posterior from a class of "homeostatic" distributions $\mathcal{Q}$: the long-term average activation of each WTA neuron z_k is constrained to an a priori defined target value. Notably, we will see that the resulting network response q^* describes an optimal variational E-Step in the sense that $q^*(\boldsymbol{z}|\boldsymbol{y}) = \arg\min_{q\in\mathcal{Q}} \mathrm{KL}\left(q(\boldsymbol{z}|\boldsymbol{y})\,||\,p(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{V})\right)$. Importantly, homeostatic plasticity fully regulates the intrinsic excitabilities, and as a side effect eliminates the non-local terms A_k in the E-step, while synaptic plasticity of the weights V_{ki} optimizes the underlying probabilistic model $p(\boldsymbol{y},\boldsymbol{z}|\boldsymbol{V})$ in the M-step.

In summary, the network response implements q^* as the variational E-step, the M-Step can be performed via gradient ascent on (6.6) with respect to V_{ki}. As derived in section 6.2.1, this gives rise to the following temporal dynamics and plasticity rules in the spiking network, which instantiate a stochastic version of the variational EM scheme:

$$u_k(t) = \sum_i V_{ki} y_i(t) + b_k \;, \qquad \dot{b}_k(t) = \eta_b \cdot \left(r_{\mathrm{net}} \cdot m_k - \delta(z_k(t) - 1)\right) \;, \tag{6.7}$$

$$\dot{V}_{ki}(t) = \eta_V \cdot \delta(z_k(t) - 1) \cdot \left(y_j(t) - \sigma(V_{ki})\right) \;, \tag{6.8}$$

where $\delta(\cdot)$ denotes the Dirac delta function, and η_b, η_V are learning rates (which were kept time-invariant in the simulations with $\eta_b = 10 \cdot \eta_V$). Note that (6.8) is a spike-timing dependent plasticity rule (cf. [159]) and is non-zero only at post-synaptic spike times t, for which $z_k(t) = 1$. The effect of the homeostatic intrinsic plasticity rule (6.7) is illustrated in Figure 6.1C: it aims to keep the long-term average activation of each WTA neuron k close to a certain target value m_k. More precisely, if r_k is a neuron's long-term average firing rate, then homeostatic plasticity will ensure that $r_k / r_{\mathrm{net}} \approx m_k$. The target activations $m_k \in (0, 1)$ can be chosen freely with the obvious constraint that $\sum_k m_k = 1$. Note that (6.7) is strongly reminiscent of homeostatic intrinsic plasticity in cortex [48, 231].

We have implemented these dynamics in a computer simulation of a WTA spiking network $\mathcal{N}$. Inputs $\boldsymbol{y}(t)$ were defined by translating handwritten digits *0-5* (Figure 6.1B) from the MNIST dataset [135] into input spike trains. Figure 6.1D shows

that, at the end of a 10^4s learning period, homeostatic plasticity has indeed achieved that $r_k \approx r_{\text{net}} \cdot m_k$. Figure 6.1E illustrates the patterns learned by each WTA neuron after this period (shown are the π_{ki}). Apparently, the WTA neurons have specialized on patterns of different intensity which correspond to different values of A_k. Figure 6.1F shows the output spiking behavior of the circuit before and after learning in response to a set of test patterns. The specialization to different patterns has led to a distinct sparse output code, in which any particular test pattern evokes output spikes from only one or two WTA neurons. Note that homeostasis forces all WTA neurons to participate in the competition, and thus prevents neurons from becoming underactive if their synaptic weights decrease, and from becoming overactive if their synaptic weights increase, much like the original A_k terms (which are nontrivial to compute for the network). Indeed, the learned synaptic parameters and the resulting output behavior corresponds to what would be expected from an optimal learning algorithm for the mixture model (6.1)-(6.3).[1]

6.2.1 Theory for the WTA model

In the following, we develop the three theoretical key results for the WTA model (6.1)-(6.3):

- Homeostatic intrinsic plasticity finds the network response distribution $q^*(\boldsymbol{z}|\boldsymbol{y}) \in \mathcal{Q}$ closest to the posterior distribution $p(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{V})$, from a set of "homeostatic" distributions $\mathcal{Q}$.
- The interplay of homeostatic and synaptic plasticity can be understood from the perspective of variational EM.
- The critical non-local terms A_k defined by (6.3) drop out of the network dynamics.

E-step: variational inference with homeostasis

The variational distribution $q(\boldsymbol{z}|\boldsymbol{y})$ we consider for the model (6.1)-(6.3) is a $2^N \cdot K$ dimensional object. Since q describes a conditional probability distribution, it is non-

[1] Without adaptation of intrinsic excitabilities, the network would start performing erroneous inference, learning would reinforce this erroneous behavior, and performance would quickly break down. We have verified this in simulations for the present WTA model: Consistently across trials, a small subset of WTA neurons became dominantly active while most neurons remained silent.

negative and normalized for all $\boldsymbol{y}$. In addition, we constrain q to be a "homeostatic" distribution $q \in \mathcal{Q}$ such that the average activation of each hidden variable (neuron) z_k equals an a-priori specified mean activation m_k under the input statistics $p^*(\boldsymbol{y})$. This is sketched in Figure 6.2. Formally we define the constraint set,

$$\mathcal{Q} = \{q : \langle z_k \rangle_{p^*(\boldsymbol{y})q(\boldsymbol{z}|\boldsymbol{y})} = m_k, \text{ for all } k = 1 \dots K\} \ , \qquad \text{with } \sum_k m_k = 1 \ . \tag{6.9}$$

The constrained maximization problem $q^*(\boldsymbol{z}|\boldsymbol{y}) = \arg\max_{q \in \mathcal{Q}} F(\boldsymbol{V}, q(\boldsymbol{z}|\boldsymbol{y}))$ can be solved with the help of Lagrange multipliers (cf. [83]). We find that the q^* which maximizes the objective function F during the E-step (and thus minimizes the KL-divergence to the posterior $p(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{V})$) has the convenient form $q^*(\boldsymbol{z}|\boldsymbol{y}) \propto p(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{V}) \cdot \exp(\sum_k \beta_k^* z_k)$ with some β_k^*. Hence, it suffices to consider distributions of the form,

$$q_{\boldsymbol{\beta}}(\boldsymbol{z}|\boldsymbol{y}) \propto \exp(\sum_k z_k (\sum_i V_{ki} y_i + \underbrace{\hat{b}_k - A_k + \beta_k}_{=: b_k})) \ , \tag{6.10}$$

for the maximization problem. We identify β_k as the variational parameters which remain to be optimized. Note that any distribution of this form can be implemented by the spiking network $\mathcal{N}$ if the intrinsic excitabilities are set to $b_k = -A_k + \hat{b}_k + \beta_k$. The optimal variational distribution $q^*(\boldsymbol{z}|\boldsymbol{y}) = q_{\boldsymbol{\beta}^*}(\boldsymbol{z}|\boldsymbol{y})$ then has $\boldsymbol{\beta}^* = \arg\max_{\boldsymbol{\beta}} \Psi(\boldsymbol{\beta})$, i.e. the variational parameter vector which maximizes the dual [83],

$$\Psi(\boldsymbol{\beta}) = \sum_k \beta_k m_k - \langle \log \sum_{\boldsymbol{z}} p(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{V}) \exp(\sum_k \beta_k z_k) \rangle_{p^*(\boldsymbol{y})} \ . \tag{6.11}$$

Due to concavity of the dual, a unique global maximizer $\boldsymbol{\beta}^*$ exists, and thus also the corresponding optimal intrinsic excitabilities $b_k^* = -A_k + \hat{b}_k + \beta_k^*$ are unique. Hence, the posterior constraint $q \in \mathcal{Q}$ can be illustrated as in Figure 6.2B: For each synaptic weight configuration $\boldsymbol{V}$ there exists, under a particular input distribution $p^*(\boldsymbol{y})$, a unique configuration of intrinsic excitabilities $\boldsymbol{b}$ such that the resulting network output fulfills the homeostatic constraints. The theoretical relation between the intrinsic excitabilities b_k, the original nontrivial term $-A_k$ and the variational parameters β_k is sketched in Figure 6.2C. Importantly, while b_k is implemented in the network, A_k, β_k and $\hat{b}_k$ are not explicitly represented in the implementation anymore. Finding the optimal $\boldsymbol{b}$ in the dual perspective, i.e. those intrinsic excitabilities which fulfill the

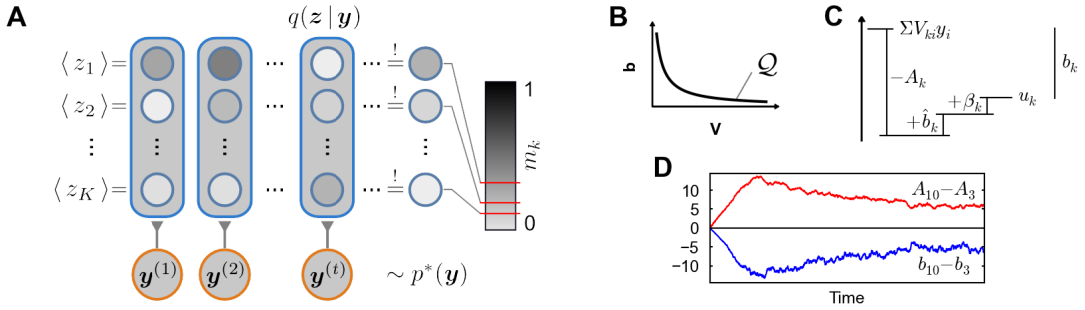

Figure 6.2: **A**. Homeostatic posterior constraints in the WTA model: Under the variational distribution q, the average activation of each variable z_k must equal m_k. **B**. For each set of synaptic weights $\boldsymbol{V}$ there exists a unique assignment of intrinsic excitabilities $\boldsymbol{b}$, such that the constraints are fulfilled. **C**. Theoretical decomposition of the intrinsic excitability b_k into $-A_k$, $\hat{b}_k$ and β_k. **D**. During variational EM the b_k predominantly "track" the dynamically changing non-local terms $-A_k$ (relative comparison between two WTA neurons from Figure 6.1).

homeostatic constraints, amounts to gradient ascent $\partial_{\boldsymbol{\beta}} \Psi(\boldsymbol{\beta})$ on the dual, which leads to the following homeostatic learning rule for the intrinsic excitabilities,

$$\Delta b_k \propto \partial_{\beta_k} \Psi(\boldsymbol{\beta}) = m_k - \langle z_k \rangle_{p^*(\boldsymbol{y}) q(\boldsymbol{z}|\boldsymbol{y})} \quad . \tag{6.12}$$

Note that the intrinsic homeostatic plasticity rule (6.7) in the network corresponds to a sample-based stochastic version of this theoretically derived adaptation mechanism (6.12). Hence, given enough time, homeostatic plasticity will automatically install near-optimal intrinsic excitabilities $\boldsymbol{b} \approx \boldsymbol{b}^*$ and implement the correct variational distribution q^* up to stochastic fluctuations in $\boldsymbol{b}$ due to the non-zero learning rate η_b. The non-local terms A_k have entirely dropped out of the network dynamics, since the intrinsic excitabilities b_k can be arbitrarily initialized, and are then fully regulated by the local homeostatic rule, which does not require knowledge of A_k.

As a side remark, note that although the variational parameters β_k are not explicitly present in the implementation, they can be theoretically recovered from the network at any point, via $\beta_k = b_k + A_k - \hat{b}_k$. Notably, in all our simulations we have consistently found small absolute values of β_k, corresponding to a small KL-divergence between q^* and p.[1] Hence, a major effect of the local homeostatic plasticity rule during learning

[1] This is assuming for simplicity uniform prior parameters $\hat{b}_k$. Note that a small KL-divergence is in fact often observed during variational EM since F, which contains the negative KL-divergence, is being maximized.

is to dynamically track and effectively implement the non-local terms $-A_k$. This is shown in Figure 6.2D, in which the relative excitabilities of two WTA neurons $b_k - b_j$ are plotted against the corresponding non-local $A_k - A_j$ over the course of learning in the first simulation (Figure 6.1).

M-step: interplay of synaptic and homeostatic intrinsic plasticity

During the M-step, we aim to increase the EM lower bound F in (6.6) w.r.t. the synaptic parameters $\boldsymbol{V}$. Gradient ascent yields,

$$\partial_{V_{ki}} F(\boldsymbol{V}, q(\boldsymbol{z}|\boldsymbol{y})) = \langle \partial_{V_{ki}} \log p(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{V}) \rangle_{p^*(\boldsymbol{y})q(\boldsymbol{z}|\boldsymbol{y})} \tag{6.13}$$

$$= \langle\, z_k \cdot (y_j - \sigma(V_{ki}))\, \rangle_{p^*(\boldsymbol{y})q(\boldsymbol{z}|\boldsymbol{y})} \quad , \tag{6.14}$$

where q is the variational distribution determined during the E-step, i.e., we can set $q = q^*$. Note the formal correspondence of (6.14) with the network synaptic learning rule (6.8). Indeed, if the network activity implements q^*, it can be shown easily that the expected update of synaptic weights due to the synaptic plasticity (6.8) is proportional to (6.14), and hence implements a stochastic version of the theoretical M-step (cf. [159]).

6.2.2 Dynamical properties of the Bayesian spiking network with homeostasis

To highlight a number of salient dynamical properties emerging from homeostatic plasticity in the considered WTA model, Figure 6.3 shows a simulation of the same network $\mathcal{N}$ with homeostatic dynamics as in Figure 6.1, only with different input statistics presented to the network, and uniform $m_k = \frac{1}{K}$. During the first 5000s, different writings of *0*'s and *3*'s from the MNIST dataset were presented, with *0*'s occurring twice as often as *3*'s. Then the input distribution $p^*(\boldsymbol{y})$ abruptly switched to include also *4*'s, with each digit occurring equally often. The following observations can be made: Due to the homeostatic constraint, each neuron responds on average to $m_k \cdot T$ out of T presented inputs. As a consequence, the number of neurons which specialize on a particular digit is directly proportional to the frequency of occurrence of that digit, i.e. 8:4 and 4:4:4 after the first and second learning period, respectively (Figure 6.3B). In general, if uniform target activations m_k are chosen, output resources are allocated precisely in proportion to input frequency. Figure 6.3C depicts the time course of the

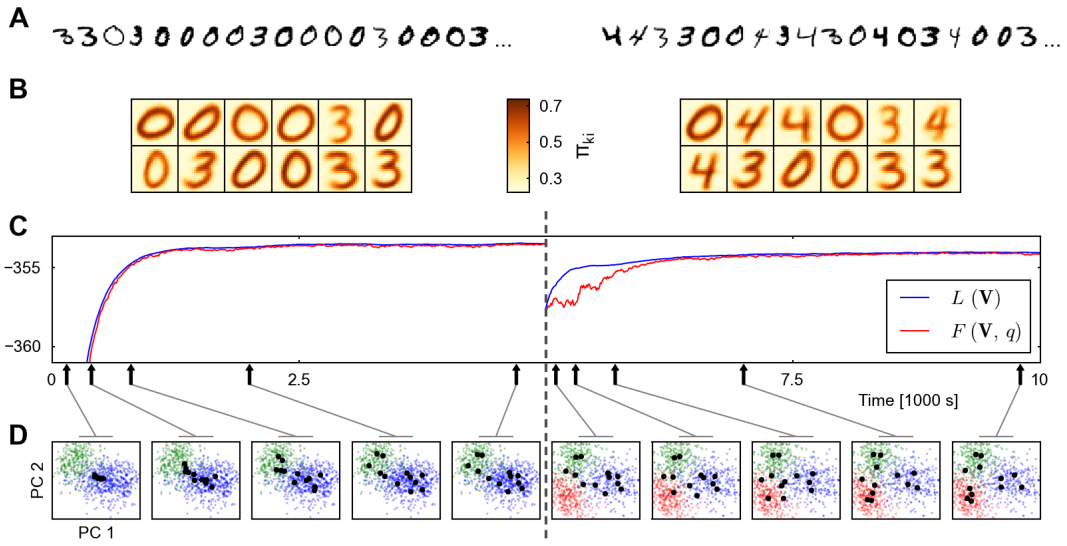

Figure 6.3: **A**. Input templates from MNIST dataset (digits *0,3* at a ratio 2:1, and digits *0,3,4* at a ratio 1:1:1) used during the first and second learning period, respectively. **B**. Learned patterns at the end of each learning period. **C**. Network performance converges in the course of learning. F is a tight lower bound to L. **D**. Illustration of pattern learning and re-learning dynamics in a 2-D projection in the input space. Each black dot corresponds to the pattern π_{ki} of one WTA neuron k. Colored dots are input samples from the training set (blue/green/red $\leftrightarrow$ digits *0/3/4*).

EM lower bound F as well as the average likelihood L (assuming uniform $\hat{b}_k$) under the model during a single simulation run, demonstrating both convergence and tightness of the lower bound. As expected due to the stabilizing dynamics of homeostasis, we found variability in performance among different trials to be small (not shown). Figure 6.3D illustrates the dynamics of learning and re-learning of patterns π_{ki} in a 2D projection of input patterns onto the first two principal components.

6.3 Homeostatic plasticity in recurrent spiking networks

The neural model so far was essentially a feed-forward network, in which every post-synaptic spike can directly be interpreted as one sample of the instantaneous posterior distribution [159]. The lateral inhibition served only to ensure the normalization of the posterior. We will now extend the concept of homeostatic processes as posterior constraints to the broader class of recurrent networks and sketch the utility of the developed framework beyond the regulation of intrinsic excitabilities.

Recently it was shown in [27, 172] that recurrent networks of stochastically spiking neurons can in principle carry out probabilistic inference through a sampling process. At every point in time, the joint network state $\boldsymbol{z}(t)$ represents one sample of a posterior. However, [27] and [172] did not consider unsupervised learning on spiking input streams.

For the following considerations, we divide the definition of the probabilistic model in two parts. First, we define a Boltzmann distribution,

$$p(\boldsymbol{z}) = \exp(\sum_k \hat{b}_k z_k + \frac{1}{2}\sum_{j\neq k} \hat{W}_{kj} z_k z_j)/\text{norm.} \ , \tag{6.15}$$

with $\hat{W}_{kj} = \hat{W}_{jk}$ as "prior" for the hidden variables $\boldsymbol{z}$ which will be represented by a recurrently connected network of K spiking neurons. For the purpose of this section, we treat $\hat{b}_k$ and $\hat{W}_{kj}$ as constants. Secondly, we define a conditional distribution in the exponential-family form [22],

$$p(\boldsymbol{y}|\boldsymbol{z}, \boldsymbol{V}) = \exp(f_0(\boldsymbol{y}) + \sum_{k,i} V_{ki} z_k f_i(\boldsymbol{y}) - A(\boldsymbol{z}, \boldsymbol{V})) \ , \tag{6.16}$$

that specifies the likelihood of observable inputs $\boldsymbol{y}$, given a certain network state $\boldsymbol{z}$. This defines the generative model $p(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{V}) = p(\boldsymbol{z})\, p(\boldsymbol{y}|\boldsymbol{z}, \boldsymbol{V})$.

We map this probabilistic model to the spiking network and define that for every k and every point in time t the variable $z_k(t)$ has the value 1, if the corresponding neuron has fired within the time window $(t-\tau, t]$. In accordance with the neural sampling theory, in order for a spiking network to sample from the correct posterior $p(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{V}) \propto p(\boldsymbol{z})\, p(\boldsymbol{y}|\boldsymbol{z}, \boldsymbol{V})$ given the input $\boldsymbol{y}$, each neuron must compute in its membrane potential the log-odd [27],

$$u_k = \log \frac{p(z_k = 1|\boldsymbol{z}_{\setminus k}, \boldsymbol{V})}{p(z_k = 0|\boldsymbol{z}_{\setminus k}, \boldsymbol{V})} = \underbrace{\sum_i V_{ki} f_i(\boldsymbol{y})}_{\text{feedforward drive}} \underbrace{-A_k(\boldsymbol{V}) + \hat{b}_k}_{\text{intr. excitability}} + \sum_{j\neq k} (\underbrace{-A_{kj}(\boldsymbol{V}) + \hat{W}_{kj}}_{\text{recurrent weight}}) z_j - \dots \tag{6.17}$$

where $\boldsymbol{z}_{\setminus k} = (z_1, \dots, z_{k-1}, z_{k+1}, \dots z_K)^{\mathsf{T}}$. The $A_k, A_{kj}, \dots$ are given by the decomposition of $A(\boldsymbol{z}, \boldsymbol{V})$ along the binary combinations of $\boldsymbol{z}$ as,

$$A(\boldsymbol{z}, \boldsymbol{V}) = A_0(\boldsymbol{V}) + \sum_k z_k A_k(\boldsymbol{V}) + \frac{1}{2}\sum_{j\neq k} z_k z_j A_{kj}(\boldsymbol{V}) + \dots \tag{6.18}$$

Note, that we do not aim at this point to give learning rules for the prior parameters $\hat{b}_k$ and $\hat{W}_{kj}$. Instead we proceed as in the last section and specify a-priori desired properties of the average network response under the input distribution $p^*(\boldsymbol{y})$,

$$c_{kj} = \langle z_k z_j \rangle_{p^*(\boldsymbol{y})q(\boldsymbol{z}|\boldsymbol{y})} \qquad \text{and} \qquad m_k = \langle z_k \rangle_{p^*(\boldsymbol{y})q(\boldsymbol{z}|\boldsymbol{y})} \ . \tag{6.19}$$

Let us explore some illustrative configurations for m_k and c_{kj}. One obvious choice is closely related to the goal of maximizing the entropy of the output code by fixing $\langle z_k \rangle$ to $\frac{1}{K}$ and $\langle z_k z_j \rangle$ to $\langle z_k \rangle \langle z_j \rangle = \frac{1}{K^2}$, thus enforcing second order correlations to be zero. Another intuitive choice would be to set all $\langle z_k z_j \rangle$ very close to zero, which excludes that two neurons can be active simultaneously and thus recovers the function of a WTA. It is further conceivable to assign positive correlation targets to groups of neurons, thereby creating populations with redundant codes. Finally, with a topographical organization of neurons in mind, all three basic ideas sketched above might be combined: one could assign positive correlations to neighboring neurons in order to create local cooperative populations, mutual exclusion at intermediate distance, and zero correlation targets between distant neurons.

With this in mind, we can formulate the goal of learning for the network in the context of EM with posterior constraints: we constrain the E-step such that the average posterior fulfills the chosen targets, and adapt the forward weights $\boldsymbol{V}$ in the M-step according to (6.6). Analogous to the first-order case, the variational solution of the E-step under these constraints takes the form,

$$q_{\boldsymbol{\beta},\boldsymbol{\omega}}(\boldsymbol{z}|\boldsymbol{y}) \propto p(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{V}) \cdot \exp\left(\sum_k \beta_k z_k + \frac{1}{2} \sum_{j \neq k} \omega_{kj} z_k z_j \right) \ , \tag{6.20}$$

with symmetric $\omega_{kl} = \omega_{lk}$ as variational parameters. A neural sampling network $\mathcal{N}$ with input weights V_{ki} will sample from $q_{\boldsymbol{\beta},\boldsymbol{\omega}}$ if the intrinsic excitabilities are set to $b_k = -A_k + \hat{b}_k + \beta_k$, and the symmetric recurrent synaptic weights to $W_{kj} = -A_{kj} + \hat{W}_{kj} + \omega_{kj}$. The variational parameters $\boldsymbol{\beta}, \boldsymbol{\omega}$ (and hence also $\boldsymbol{b}, \boldsymbol{W}$) which optimize the dual problem $\Psi(\boldsymbol{b}, \boldsymbol{\omega})$ are uniquely defined and can be found iteratively via gradient ascent. Analogous to the last section, this yields the intrinsic plasticity rule (6.12) for b_k. In addition, we obtain for the recurrent synapses W_{kj},

$$\Delta W_{kj} \propto c_{kj} - \langle z_k z_j \rangle_{p^*(\boldsymbol{y})q(\boldsymbol{z}|\boldsymbol{y})} \ , \tag{6.21}$$

which translates to an anti-Hebbian spike-timing dependent plasticity rule in the network implementation.

For any concrete instantiation of $f_0(\boldsymbol{y})$, $f_i(\boldsymbol{y})$ and $A(\boldsymbol{z}, \boldsymbol{V})$ in (6.16) it is possible to derive learning rules for V_{ki} for the M-step via $\partial_{V_{ki}} F(\boldsymbol{V}, q)$. Of course not all models entail local synaptic learning rules. In particular it might be necessary to assume conditional independence of the inputs $\boldsymbol{y}$ given the network state $\boldsymbol{z}$, i.e., $p(\boldsymbol{y}|\boldsymbol{z}, \boldsymbol{V}) = \prod_i p(y_i|\boldsymbol{z}, \boldsymbol{V})$. Furthermore, in order to fulfill the neural computability condition (6.17) for neural sampling [27] with a recurrent network of point neurons, it might be necessary to choose $A(\boldsymbol{z}, \boldsymbol{V})$ such that terms of order higher than 2 vanish in the decomposition. This can be shown to hold, for example, in a model with conditionally independent Gaussian distributed inputs y_i. It is ongoing work to find further biologically realistic network models in the sense of this theory and to assess their computational capabilities through computer experiments.

6.4 Discussion

Complex and non-local computations, which appear during probabilistic inference and learning, arguably constitute one of the cardinal challenges in the development of biologically realistic Bayesian spiking network models. In this paper we have introduced homeostatic plasticity, which to the best of our knowledge had not been considered before in the context of EM in spiking networks, as a theoretically grounded approach to stabilize and facilitate learning in a large class of network models. Our theory complements previously proposed neural mechanisms and provides, in particular, a simple and biologically realistic alternative to the assumptions on the input distribution made in [159] and [116]. Indeed, our results challenge the hypothesis of [116] that feedforward inhibition is critical for correctly learning the structure of the data with biologically plausible plasticity rules. More generally, it turns out that the enforcement of a balancing posterior constraint often simplifies inference in recurrent spiking networks by eliminating nontrivial computations. Our results suggest a crucial role of homeostatic plasticity in the Bayesian brain: to constrain activity patterns in cortex to assist the autonomous optimization of an internal model of the environment.

References

[1] L. F. Abbott S. B. Nelson. **Synaptic plasticity: taming the beast**. *Nature Neuroscience*, **3**:1178–1183, 2000. 3, 28, 73, 82, 102

[2] W. C. Abraham. **Metaplasiticity: tuning synapses and networks for plasticity**. *Nature Reviews Neuroscience*, **9**:387–399, 2008. 49

[3] D. H. Ackley, G. E. Hinton, T. J. Sejnowski. **A Learning Algorithm for Boltzmann Machines**. *Cognitive Science*, **9**:147–169, 1985. 152, 157, 175

[4] D. Alais R. Blake. *Binocular Rivalry*. MIT Press, 2005. 170, 175, 176

[5] C. Andrieu, N. D. Freitas, A. Doucet, M. I. Jordan. **An Introduction to MCMC for Machine Learning**. *Machine Learning*, **50**:5–43, 2003. 154, 160

[6] D. E. Angelaki, Y. Gu, G. C. DeAngelis. **Multisensory integration: psychophysics, neurophysiology and computation**. *Current opinion in neurobiology*, **19**(4):452–458, 2009. 204

[7] S. D. Antic, W. L. Zhou, A. R. Moore, S. M. Short, K. D. Ikonomu. **The decade of the dendritic NMDA spike**. *J Neurosci Res*, **88**(14):2991–3001, 2010. 176

[8] H. Attias. **Planning by probabilistic inference**. In *Proc. of the 9th Int. Workshop on Artificial Intelligence and Statistics*, 2003. 61

[9] J.-Y. Audibert, R. Munos, C. Szepesvari. **Tuning bandit problems in stochastic environments**. In *Proc. of the 18th International Conference on Algorithmic Learning Theory*, pages 150–165, 2007. 61

[10] P. Auer, N. Cesa-Bianchi, P. Fischer. **Finite Time Analysis of the Multiarmed Bandit Problem**. *Machine Learning*, **47**(2/3):235–256, 2002. 31

[11] P. Auer, T. Jaksch, R. Ortner. **Near-optimal regret bounds for reinforcement learning**. In *Advances in Neural Information Processing Systems 21*, pages 89–96, Cambridge, MA, 2009. MIT Press. 61

[12] F. A. C. Azevedo, L. R. B. Carvalho, L. T. Grinberg, et al. **Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain.** *The Journal of comparative neurology*, **513**(5):532–41, April 2009. 4

[13] R. Azouz C. M. Gray. **Cellular mechanisms contributing to response variability of cortical neurons in vivo**. *J. Neuroscience*, **19**:2209–2223, 1999. 150

[14] C. H. Bailey, M. Giustetto, Y.-Y. Huang, R. D. Hawkins, E. R. Kandel. **Is heterosynaptic modulation essential for stabilizing Hebbian plasticity and memory?** *Nature Reviews Neuroscience*, **1**:11–20, 2000. 28, 55, 59

REFERENCES

[15] A. Bartels N. K. Logothetis. **Binocular rivalry: a time dependence of eye and stimulus contributions**. *J. of Vision*, **20**(12):article 3, 2010. 170, 175, 176

[16] P. Berkes, G. Orban, M. Lengyel, J. Fiser. **Spontaneous Cortical Activity Reveals Hallmarks of an Optimal Internal Model of the Environment**. *Science*, **331**:83–87, 2011. 175, 176, 204

[17] D. P. Bertsekas J. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA, 1996. 15, 61

[18] G. Bi M. Poo. **Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type**. *J Neuroscience*, **18**(24):10464–10472, 1998. 81, 101, 118

[19] T. Binzegger, R. J. Douglas, K. A. Martin. **A quantitative map of the circuit of cat primary visual cortex**. *J. Neurosci.*, **24**(39):8441–8453, 2004. 173

[20] T. Binzegger, R. J. Douglas, K. A. Martin. **Stereotypical bouton clustering of individual neurons in cat primary visual cortex**. *J. Neurosci.*, **27**(45):12242–12254, 2007. 121

[21] T. Binzegger, R. Douglas, K. Martin. **Topology and dynamics of the canonical circuit of cat V1**. *Neural Networks*, **22**(8):1071 – 1078, 2009. Cortical Microcircuits. 173

[22] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, 2006. 12, 14, 20, 29, 30, 42, 62, 75, 92, 175, 207, 214

[23] R. Blake N. K. Logothetis. **Visual competition**. *Nature Reviews Neuroscience*, **3**:13–21, 2002. 170, 175, 176

[24] J. W. Brascamp, R. van Ee, A. J. Noest, R. H. A. H. Jacobs, A. V. van den Berg. **The time course of binocular rivalry reveals a fundamental role of noise**. *Journal of Vision*, **6**:1244–1256, 2006. 150

[25] J. Brea, W. Senn, J.-P. Pfister. **Sequence learning with hidden units in spiking neural networks**. In *Proc. of NIPS 2011*, **24**, pages 1422–1430. MIT Press, 2012. 118, 204

[26] D. Bruederle, J. Bill, B. Kaplan, et al. **Live demonstration: Simulator-like exploration of cortical network architectures with a mixed-signal VLSI system**. In *Proc. of the IEEE Int. Symp. on Circuits and Systems*, page 2783, 2010. 177

[27] L. Büsing, J. Bill, B. Nessler, W. Maass. **Neural Dynamics as Sampling: A Model for Stochastic Computation in Recurrent Networks of Spiking Neurons**. *PLoS Computational Biology*, 2011. doi:10.1371/journal.pcbi.1002211. 8, 74, 113, 204, 214, 216

[28] R. Cannon, C. O'Donnell, M. Nolan. **Stochastic ion channel gating in dendritic neurons: morphology dependence and probabilistic synaptic activation of dendritic spikes**. *PLoS Comput Biol*, **6**(8):e1000886, 2010. 150

[29] N. Caporale Y. Dan. **Spike timing-dependent plasticity: a Hebbian learning rule**. *Annu Rev Neuroscience*, **31**:25–46, 2008. 28, 80, 120

[30] M. Carandini D. Heeger. **Normalization as a canonical neural computation**. *Nature Reviews Neuroscience*, **13**:51–62, 2012. 73, 75

[31] G. Celeux J. Diebolt. **The SEM algorithm: A probabilistic teacher algorithm derived from the EM algorithm for the mixture problem**. *Comput. Statist. Quater.*, **2**:73–82, 1985. 115

[32] M. M. Churchland, B. M. Yu, J. P. Cunningham, et al. **Stimulus onset quenches neural variability: a widespread cortical phenomenon**. *Nature Neuroscience*, **13**(3):369–378, 2010. 174

[33] D. C. Ciresan, U. Meier, L. M. Gambardella, J. Schmidhuber. **Deep, big, simple neural nets for handwritten digit recognition.** *Neural computation*, **22**(12):3207–20, 2010. 107

[34] C. Clopath, L. Büsing, E. Vasilaki, W. Gerstner. **Connectivity reflects coding: a model of voltage-based STDP with homeostasis**. *Nature Neuroscience*, **13**(3):344–352, 2010. 101

[35] R. H. Cudmore G. G. Turrigiano. **Long-term potentiation of intrinsic excitability in LV visual cortical neurons**. *J. Neurophysiol.*, **92**(1):341–348, 2004. 82, 120

[36] Y. Dan M. Poo. **Spike Timing-Dependent Plasticity of Neural Circuits**. *Neuron*, **44**:23–30, 2004. 73, 80

[37] G. Daoudal D. Debanne. **Long-term plasticity of intrinsic excitability: learning rules and mechanisms**. *Learn. Mem.*, **10**(6):456–465, 2003. 73, 82, 120

[38] P. Dayan L. F. Abbott. *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. MIT Press, Cambridge, MA, 2001. 2, 28, 60, 116

[39] P. Dayan N. Daw. **Decision theory, reinforcement learning, and the brain**. *Cognitive, Affective, & Behavioral Neuroscience*, **8**:429–453, 2008. 31

[40] P. Dayan S. Kakade. **Explaining away in weight space**. In *Advances in Neural Information Processing Systems 13*, pages 451–457, Cambridge, MA, 2001. MIT Press. 55, 61

[41] P. Dayan A. Yu. **Uncertainty and Learning**. *IETE Journal of Research*, **49**:171–182, 2003. 61

[42] E. de Villers-Sidani M. M. Merzenich. **Lifelong plasticity in the rat auditory cortex: basic mechanisms and role of sensory experience**. *Prog Brain Res*, **191**:119–131, 2011. 121

[43] D. Debanne M.-M. Poo. **Spike-timing dependent plasticity beyond synapse – pre- and post-synaptic plasticity of intrinsic neuronal excitability**. *Front. Syn. Neurosci*, **2**(21), 2010. doi: 10.3389/fnsyn.2010.00021. 120

[44] A. P. Dempster, N. M. Laird, D. B. Rubin. **Maximum Likelihood from Incomplete Data via the EM Algorithm**. *Journal of the Royal Statistical Society. Series B (Methodological)*, **39**(1):1–38, 1977. 75, 90, 92, 115

[45] S. Deneve. **Bayesian spiking neurons I: Inference**. *Neural Computation*, **20**(1):91–117, 2008. 12, 42, 62, 117, 152, 176, 204

[46] S. Deneve. **Bayesian spiking neurons II: Learning**. *Neural Computation*, **20**(1):118–145, 2008. 12, 117, 204, 208

[47] S. Denison, E. Bonawitz, A. Gopnik, T. Griffiths. **Preschoolers sample from probability distributions**. In *Proc. of the 32nd Annual Conference of the Cognitive Science Society*, 2010. 175

[48] N. Desai, L. Rutherford, G. Turrigiano. **Plasticity in the intrinsic excitability of cortical pyramidal neurons**. *Nature Neuroscience*, **2**(6):515, 1999. 205, 208

[49] R. Desimone. **Face-Selective Cells in the Temporal Cortex of Monkeys**. *Journal of Cognitive Neuroscience*, **3**(1):1–8, 1991. 114

REFERENCES

[50] A. Destexhe, M. Rudolph, J. M. Fellous, T. J. Sejnowski. **Fluctuating synaptic conductances recreate in vivo-like activity in neocortical neurons**. *Neuroscience*, **107**(1):13–24, 2001. 121, 137

[51] P. Domingos M. Pazzani. **On the optimality of the simple Bayesian classifier under zero-one loss**. *Machine Learning*, **275**(29):103–130, 1997. 29

[52] R. J. Douglas K. A. Martin. **Neuronal circuits of the neocortex**. *Annu Rev Neurosci*, **27**:419–451, 2004. 3, 57, 73, 77, 97, 121

[53] K. Doya. **Metalearning and neuromodulation**. *Neural Networks*, **15**:495–506, 2002. 49

[54] K. Doya, S. Ishii, A. Pouget, R. P. N. Rao. *Bayesian Brain: Probabilistic Approaches to Neural Coding*. MIT-Press, 2007. 72, 113, 151

[55] V. Dragalin, A. Tartakovsky, V. Veeravalli. **Multihypothesis Sequential Probability Ratio Tests — Part I: Asymptotic Optimality**. *IEEE Transactions on Information Theory*, **45**(7):2448–2461, 1999. 57

[56] A. S. Ecker, P. Berens, G. A. Keliris, et al. **Decorrelated neuronal firing in cortical microcircuits**. *Science*, **327**:584–587, 2010. 77, 120

[57] J. S. Espinosa M. P. Stryker. **Development and Plasticity of the Primary Visual Cortex**. *Neuron*, **75**(2):230–249, 2012. 106

[58] A. A. Faisal, L. P. J. Selen, D. M. Wolpert. **Noise in the nervous system**. *Nature Reviews Neuroscience*, **9**(4):292–303, 2008. 120

[59] M. A. Farries A. L. Fairhall. **Reinforcement Learning with Modulated Spike Timing-Dependent Synaptic Plasticity**. *Journal of Neurophysiology*, **98**:3648–3665, 2007. 28

[60] D. Feldman. **The spike-timing dependence of plasticity**. *Neuron*, **75**(4):556–571, 2012. 73

[61] E. Fino R. Yuste. **Dense Inhibitory Connectivity in Neocortex**. *Neuron*, **69**(6):1188–1203, 2011. 77

[62] J. Fiser, P. Berkes, G. Orban, M. Lengyel. **Statistically optimal perception and learning: from behavior to neural representation**. *Trends in Cogn. Sciences*, **14**(3):119–130, 2010. 72, 73, 152, 175, 204

[63] J. Fiser, C. Chiu, M. Weliky. **Small modulation of ongoing cortical dynamics by sensory input during natural vision**. *Nature*, **431**:573–583, 2004. 150, 173

[64] M. Flight. **Synaptic transmission: On the probability of release**. *Nature Reviews Neuroscience*, **9**:736–737, 2010. 150

[65] M. D. Fox M. E. Raichle. **Spontaneous fluctuations in brain activity observed with functional magnetic resonance imaging**. *Nature Reviews Neuroscience*, **8**:700–711, 2007. 176

[66] Y. Fregnac. **Hebbian synaptic plasticity**. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 515–522. MIT Press, Cambridge, MA, 2003. 28

[67] K. J. Friston, J. Daunizeau, J. Kilner, S. J. Kiebel. **Action and behavior: a free-energy formulation**. *Biol Cybern*, **102**(3):227–260, 2010. 151

[68] R. C. Froemke Y. Dan. **Spike-timing-dependent synaptic modification induced by natural spike trains**. *Nature*, **415**:433–438, 2002. 103, 104

[69] C. Gardiner. *Handbook of Stochastic Methods*. 3rd ed. Springer, 2004. 168

[70] S. Geman D. Geman. **Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **6**(6):721–741, 1984. 151, 153

[71] A. P. Georgopoulos, A. P. Schwartz, R. E. Ketner. **Neuronal population coding of movement direction**. *Science*, **233**:1416–1419, 1986. 51, 58

[72] S. J. Gershman, E. Vul., J. Tenenbaum. **Perceptual Multistability as Markov Chain Monte Carlo Inference**. *Advances in Neural Information Processing Systems*, **22**:611–619, 2009. 151, 152, 153, 170, 174, 176

[73] W. Gerstner W. M. Kistler. *Spiking Neuron Models*. Cambridge University Press, Cambridge, 2002. 78, 98, 150, 162, 167

[74] Z. Ghahramani M. I. J. Michael. **Mixture models for learning from incomplete data**. In *Computational Learning Theory and Natural Learning Systems: Volume IV: Making Learning Systems Practical*, pages 67–85. MIT Press Cambridge, MA, USA, 1997. 87, 133

[75] C. D. Gilbert, W. Li, V. Piech. **Perceptual learning and adult cortical plasticity**. *J. Physiol.*, **387**:2743–2751, 2009. 120

[76] C. D. Gilbert, M. Sigman, R. E. Crist. **The neural basis of perceptual learning**. *Neuron*, **31**(5):681–697, 2001. 120

[77] M. Gilson, T. Masquelier, E. Hugues. **STDP Allows Fast Rate-Modulated Coding with Poisson-Like Spike Trains**. *PLoS Computational Biology*, **7**(10):e1002231, 2011. 87, 102, 111, 116

[78] J. Gittins. **Bandit processes and dynamic allocation indices**. *Journal of the Royal Statistical Society*, **41**:148–177, 1979. 31

[79] J. Gjorgjieva, C. Clopath, J. Audet, J.-P. Pfister. **A triplet spike-timing-dependent plasticity model generalizes the Bienenstock-Cooper-Munro rule to higher-order spatiotemporal correlations**. *PNAS*, **108**(48):19383–19388, 2011. 100, 101

[80] M. Goard Y. Dan. **Basal forebrain activation enhances cortical coding of natural scenes**. *Nature Neuroscience*, **12**(11):1444–1449, 2009. 120

[81] J. I. Gold M. N. Shadlen. **The neural basis of decision making**. *Annu Rev Neuroscience*, **30**:535–574, 2007. 57, 151, 161

[82] A. Gopnik J. B. Tenenbaum. **Bayesian special section: Introduction; Bayesian networks, Bayesian learning and cognitive development**. *Developmental Science*, **10**(3):281–287, 2007. 151

[83] J. Graca, K. Ganchev, B. Taskar. **Expectation maximization and posterior constraints**. In *Proc. of NIPS 2007*, **20**. MIT Press, 2008. 205, 210

[84] M. Graupner N. Brunel. **Calcium-based plasticity model explains sensitivity of synaptic changes to spike pattern, rate, and dendritic location**. *PNAS*, **109**(10):3991–3996, 2012. 101

[85] T. Griffiths J. Tenenbaum. **Structure and strength in causal induction**. *Cognitive Psychology*, **51**:334–384, 2005. 61

REFERENCES

[86] T. L. Griffiths, C. Kemp, J. B. Tenenbaum. **Bayesian models of cognition**. In R. Sun, editor, *Handbook of Computational Cognitive Modeling*, chapter 3, page 59–100. Cambridge Univ. Press, 2008. 72, 113, 151

[87] T. L. Griffiths J. B. Tenenbaum. **Optimal predictions in everyday cognition**. *Psychological Science*, **17**(9):767–773, 2006. 5, 72, 113, 151, 175, 204

[88] S. Grillner A. Graybiel. *Microcircuits: The Interface between Neurons and Global Brain Function*. MIT-Press, 2006. 73

[89] G. R. Grimmett D. R. Stirzaker. *Probability and Random Processes*. Oxford University Press, 3rd edition, 2001. 154

[90] A. Gupta L. N. Long. **Character Recognition using Spiking Neural Networks**. *IJCNN*, pages 53–58, 2007. 116

[91] A. Gupta L. N. Long. **Hebbian Learning with Winner Take All for Spiking Neural Networks**. *IEEE International Joint Conference on Neural Networks*, pages 1189–1195, 2009. 116

[92] K. Gurney R. Bogacz. **The basal ganglia and cortex implement optimal decision making between alternative actions**. *Neural Computation*, **19**:442–477, 2006. 57, 60

[93] R. Gütig H. Sompolinsky. **Time-Warp Invariant Neuronal Processing**. *PLoS Biology*, **7**(7):e1000141, 2009. 117

[94] S. Habenschuss, J. Bill, B. Nessler. **Homeostatic plasticity in Bayesian spiking networks as Expectation Maximization with posterior constraints**. In *Advances in Neural Information Processing Systems 25*, pages 782–790, 2012. 8, 82

[95] R. H. R. Hahnloser, R. Sarpeshkar, M. A. Mahowald, R. J. Douglas, H. S. Seung. **Digital Selection and Analogue Amplification coexist in a Cortex-Inspired Silicon Circuit**. *Nature*, **405**:947–951, 2000. 57, 73

[96] D. O. Hebb. *The Organization of Behavior*. Wiley, New York, 1949. 2, 12, 28, 81

[97] G. E. Hinton. **Training Products of Experts by Minimizing Contrastive Divergence**. *Neural Computation*, **14**(8):1771–1800, 2002. 191

[98] G. E. Hinton. **Learning to represent visual input**. *Philosophical Transactions of the Royal Society, B*, **365**:177–184, 2010. 157, 191

[99] G. E. Hinton Z. Ghahramani. **Generative models for discovering sparse distributed representations**. *Philos Trans R Soc Lond B Biol Sci.*, **352**(1358):1177–1190, 1997. 74

[100] G. E. Hinton, S. Osindero, Y.-W. Teh. **A Fast Learning Algorithm for Deep Belief Nets**. *Neural Computation*, **18**:1527–1554, 2006. 74, 107, 153, 157

[101] G. E. Hinton R. R. Salakhutdinov. **Reducing the dimensionality of data with neural networks**. *Science*, **313**(5786):504–507, 2006. 107, 114

[102] G. Hinton A. Brown. **Spiking Boltzmann machines**. In *Proceedings of the 13th Conference on Advances in Neural Information Processing Systems; December 1999; Vancouver, Canada. NIPS 1999*, 2000. 175

[103] J. J. Hopfield D. W. Tank. **"Neural" Computation of Decisions in Optimization Problems**. *Biological Cybernetics*, **52**:141–152, 1985. 175

[104] P. Hoyer A. Hyvärinen. **Interpreting neural response variability as Monte Carlo sampling of the posterior**. In *Proceedings of the 16th Conference on Advances in Neural Information Processing Systems; December 2002; Vancouver, Canada. NIPS 2002.*, 2003. 5, 8, 151, 152, 170, 176

[105] D. H. Hubel T. N. Wiesel. **Receptive fields, binocular interaction and functional architecture in the cat's visual cortex**. *The Journal of physiology*, **160**(1):106, 1962. 3

[106] J. S. Ide F. G. Cozman. **Random Generation of Bayesian Networks**. In *Proc. of the 16th Brazilian Symposium on Artificial Intelligence: Advances in Artificial Intelligence*, **2507**, pages 366–375, London, 2002. Springer. 20, 21, 49

[107] G. Indiveri, B. Linares-Barranco, T. Hamilton, et al. **Neuromorphic silicon neuron circuits**. *Frontiers in Neuroscience*, **5**:1–23, 2011. 115

[108] L. Itti, C. Koch, E. Niebur. **A Model of Saliency-Based Visual Attention for Rapid Scene Analysis**. *IEEE Trans. Pattern Anal. Mach. Intell.*, **20**(11):1254–1259, 1998. 73

[109] E. M. Izhikevich. **Solving the Distal Reward Problem through Linkage of STDP and Dopamine Signaling**. *Cerebral Cortex*, **17**:2443–2452, 2007. 3

[110] W. Jank. **The EM algorithm, its randomized implementation and global optimization: Some challenges and opportunities for operations research**. *Perspectives in Operations Research*, pages 367–392, 2006. 92

[111] F. V. Jensen T. D. Nielsen. *Bayesian Networks and Decision Graphs (2nd edition)*. Springer, New York, 2007. 58

[112] X. Jin, M. Lujan, L. Plana, et al. **Modeling Spiking Neural Networks on SpiNNaker**. *Computing in Science & Engineering*, **12**(5):91–97, September-October 2010. 115

[113] R. Jolivet, A. Rauch, H. Lüscher, W. Gerstner. **Predicting spike timing of neocortical pyramidal neurons by simple threshold models**. *Journal of Computational Neuroscience*, **21**:35–49, 2006. 78, 113, 206

[114] M. I. Jordan R. A. Jacobs. **Hierarchical Mixtures of Experts and the Algorithm**. *Neural Computation*, **6**:181–214, 1994. 115

[115] M. Kearns S. Singh. **Near-optimal performance for reinforcement learning in polynomial time**. In *Proc. of the 15th International Conference on Machine Learning (ICML)*, pages 260–268, 1998. 61

[116] C. Keck, C. Savin, J. Lücke. **Feedforward Inhibition and Synaptic Scaling - Two Sides of the Same Coin?** *PLoS Computational Biology*, **8**(3):e1002432, 2012. 74, 204, 206, 207, 208, 216

[117] R. Kempter, W. Gerstner, J. L. van Hemmen. **Hebbian Learning and Spiking Neurons**. *Phys. Rev. E*, **59**(4):4498–4514, 1999. 73

[118] R. Kempter, W. Gerstner, J. L. van Hemmen. **Intrinsic stabilization of output rates by spike-based Hebbian learning**. *Neural Computation*, **13**:2709–2741, 2001. 73

[119] T. Kenet, D. Bibitchkov, M. Tsodyks, A. Grinvald, A. Arieli. **Spontaneously emerging cortical representations of visual attributes**. *Nature*, **425**(6961):954–956, 2003. 176

[120] J. N. D. Kerr, C. P. J. de Kock, D. S. Greenberg, et al. **Spatial organization of neuronal population responses in layer 2/3 of rat barrel cortex**. *J Neuroscience*, **27**(48):13316–13328, 2007. 120

REFERENCES

[121] D. Kersten, P. Mamassian, A. Yuille. **Object perception as Bayesian inference**. *Annual Review of Psychology*, **55**(1):271–304, 2004. 151

[122] D. C. Knill W. Richards. *Perception as Bayesian Inference*. Cambridge Universiy Press, 1996. 5

[123] D. C. Knill. **Discrimination of planar surface slant from texture: human and ideal observers compared**. *Vision Research*, **38**(11):1683–1711, 1998. 5

[124] K. Kobayashi M. Poo. **Spike Train Timing-Dependent Associative Modification of Hippocampal CA3 Recurrent Synapses by Mossy Fibers**. *Neuron*, **41**:445–454, 2004. 101, 102

[125] D. Koller N. Friedman. *Probabilistic Graphical Models: Principles and Techniques (Adaptive Computation and Machine Learning)*. MIT Press, 2009. 175

[126] I. Kononenko. **Bayesian neural networks**. *Biol. Cybernetics*, **61**:361–370, 1998. 62

[127] K. P. Körding D. M. Wolpert. **Bayesian integration in sensorimotor learning**. *Nature*, **427**(6971):244–247, 2004. 72, 113, 151, 204

[128] K. Körding. **Decision theory: what "should" the nervous system do?** *Science (New York, N.Y.)*, **318**(5850):606–10, October 2007. 5

[129] K. P. Körding D. M. Wolpert. **Bayesian decision theory in sensorimotor control**. *Trends in Cognitive Sciences*, **10**(7):319–326, 2006. 5

[130] F. R. Kschischang, B. J. Frey, H. A. Loeliger. **Factor graphs and the sum-product algorithm**. *IEEE Transactions on Information Theory*, **47**(2):498–519, 2001. 29, 30, 42, 58

[131] H. Kushner G. Yin. *Stochastic approximation and recursive algorithms and applications*, **35**. Springer Verlag, 2003. 95, 129

[132] T. Lai H. Robbins. **Asymptotically efficient adaptive allocation rules**. *Advances in Applied Mathematics*, **6**:4–22, 1985. 31

[133] A. Lansner O. Ekeberg. **A one-layer feedback artificial neural network with a Bayesian learning rule**. *International Journal of Neural Systems*, **1**:77–87, 1998. 62

[134] A. Lansner A. Holst. **A Higher Order Bayesian Neural Network with spiking units**. *International Journal of Neural Systems*, **7**(2):115–128, 1996. 62

[135] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner. **Gradient-based learning applied to document recognition**. *Proceedings of the IEEE*, **86**(11):2278–2324, 11 1998. 107, 208

[136] T. S. Lee D. Mumford. **Hierarchical Bayesian inference in the visual cortex**. *J. Opt. Soc. Am. A*, **20**(7):1434–1448, 2003. 151

[137] R. Legenstein, D. Pecevski, W. Maass. **A Learning Theory for Reward-Modulated Spike-Timing-Dependent Plasticity with Application to Biofeedback**. *PLoS Computational Biology*, **4**(10):1–27, 2008. 28

[138] D. A. Leopold, M. Wilke, A. Maier, N. K. Logothetis. **Stable perception of visually ambiguous patterns**. *Nature Neuroscience*, **5**:605–609, 2002. 170, 175, 176

[139] Y. Li, D. Fitzpatrick, L. White. **The development of direction selectivity in ferret visual cortex requires early visual experience**. *Nature neuroscience*, **9**(5):676–681, 2006. 106

[140] D. Liao, A. Jones, R. Malinow. **Direct measurement of quantal changes underlying long-term potentiation in CA1 hippocampus**. *Neuron*, **9**(6):1089–1097, 1992. 118

[141] S. Litvak S. Ullman. **Cortical Circuitry Implementing Graphical Models**. *Neural Comput*, **21**:1–47, 2009. 152

[142] W. Lohmiller J. J. Slotine. **Contraction analysis for nonlinear systems**. *Automatica*, **34**(6):683–696, 1998. 56, 67

[143] W. Ma, J. Beck, P. Latham, A. Pouget. **Bayesian inference with probabilistic population codes**. *Nature neuroscience*, 2006. 5, 73, 117

[144] W. Maass. **On the computational power of winner-take-all**. *Neural Computation*, **12**(11):2519–2535, 2000. 27, 73, 74

[145] P. Maldonado, C. Babul, W. Singer, et al. **Synchronization of neuronal responses in primary visual cortex of monkeys viewing natural images**. *J Neurophysiol*, **100**(3):1523–1532, 2008. 121

[146] N. T. Markov, P. Misery, A. Falchier, et al. **Weight Consistency Specifies Regularities of Macaque Cortical Networks**. *Cerebral Cortex*, **21**(6):1254–1272, 2011. 121

[147] T. Masquelier, R. Guyonneau, S. Thorpe. **Spike timing dependent plasticity finds the start of repeating patterns in continuous spike trains**. *PLoS ONE*, **3**(1), 2008. 102, 111, 116

[148] T. Masquelier, R. Guyonneau, S. Thorpe. **Competitive STDP-based spike pattern learning**. *Neural Computation*, **21**(5):1259–1276, 2009. 111, 117

[149] T. Masquelier, E. Hugues, G. Deco, S. Thorpe. **Oscillations, Phase-of-Firing Coding, and Spike Timing-Dependent Plasticity:An Efficient Learning Scheme**. *Journal of Neuroscience*, **29**(43):13484–13493, 2009. 87, 102, 116

[150] L. B. Merabet A. Pascual-Leone. **Neural reorganization following sensory loss: the opportunity of change**. *Nature Reviews Neuroscience*, **11**:44–52, 2010. doi: 10.1038/nrn2758. 121

[151] P. Merolla, J. Arthur, B. E. Shi, K. Boahen. **Expandable Networks for Neuromorphic Chips**. *IEEE Transactions on Circuits and Systems I*, **54**(2):301–311, 2007. 177

[152] P. Montague, P. Dayan, C. Person, T. Sejnowski. **Bee foraging in uncertain environments using predictive Hebbian learning**. *Nature*, **377**:725–728, 1995. 60

[153] J. M. Montgomery, P. Pavlidis, D. V. Madison. **Pair recordings reveal all-silent synaptic connections and the postsynaptic expression of long-term potentiation**. *Neuron*, **29**(3):691–701, 2001. 118

[154] A. Morrison, M. Diesmann, W. Gerstner. **Phenomenological models of synaptic plasticity based on spike timing**. *Biological Cybernetics*, **98**(6):459–478, 2008. 101

[155] R. M. Neal G. E. Hinton. **A view of the EM algorithm that justifies incremental sparse, and other variants**. In M. I. Jordan, editor, *Learning in Graphical Models*. Kluwer Academic Press, 1998. 115

[156] R. Neapolitan. *Learning Bayesian Networks*. Prentice Hall, Upper Saddle River, NJ, 2004. 34, 58

[157] E. Neftci, E. Chicca, G. Indiveri, J. Slotine, R. Douglas. **Contraction Properties of VLSI Cooperative Competitive Neural Networks of Spiking Neurons**. In *Advances in Neural Information Processing Systems*, Cambridge, MA, 2008. MIT Press. 57

[158] B. Nessler, M. Pfeiffer, W. Maass. **Hebbian learning of Bayes optimal decisions**. *In Proc. of NIPS 2008: Advances in Neural Information Processing Systems*, **21**, 2009. MIT Press. 6, 47, 48, 73, 114, 134, 157

[159] B. Nessler, M. Pfeiffer, W. Maass. **STDP enables spiking neurons to detect hidden causes of their inputs**. In *Proc. of NIPS 2009: Advances in Neural Information Processing Systems*, **22**, pages 1357–1365. MIT Press, 2010. 7, 76, 204, 205, 206, 208, 212, 213, 216

[160] A. Y. Ng M. I. Jordan. **On discriminative vs. generative classifiers**. *NIPS*, **14**:841–848, 2002. 23

[161] D. Nikolic, S. Haeusler, W. Singer, W. Maass. **Distributed fading memory for stimulus properties in the primary visual cortex**. *PLoS Biology*, **7**(12):1–19, 2009. 120

[162] S. J. Nowlan. **Maximum Likelihood Competitive Learning**. In D. Touretzky, editor, *Advances in Neural Information Processing Systems (NIPS)*, **2**, pages 574–582. Morgan Kaufmann, San Mateo, California, 1990. 115

[163] S. J. Nowlan. **Soft competitive adaptation: neural network learning algorithms based on fitting statistical mixtures**. Technical Report CS-91-126, Carnegie Mellon University, Pittsburgh, 1991. 115

[164] M. Oaksford N. Chater. *Bayesian Rationality: The Probabilistic Approach to Human Reasoning*. Oxford University Press, 2007. 113, 151

[165] J. O'Keefe, N. Burgess, J. Donnett, K. Jeffery, E. Maguire. **Place cells, navigational accuracy, and the human hippocampus**. *Philosophical Transactions of the Royal Society of London*, **353**(1373):1333–1340, 1998. 51, 58

[166] M. Okun I. Lampl. **Instantaneous correlation of excitation and inhibition during ongoing and sensory-evoked activities**. *Nature Neuroscience*, **11**(5):535–537, 2008. 77, 120

[167] B. A. Olshausen D. J. Field. **Emergence of simple-cell receptive field properties by learning a sparse code for natural images**. *Nature*, **381**:607–609, 1996. 58, 106, 107

[168] B. A. Olshausen D. J. Field. **How close are we to understanding V1?** *Neural Computation*, **17**(8):1665–1699, 2005. 120

[169] G. Orban, J. Fiser, R. Aslin, M. Lengyel. **Bayesian learning of visual chunks by human observers**. *Proceedings of the National Academy of Sciences*, **105**(7):2745–2750, 2008. 5, 204

[170] M. Oster, R. Douglas, S. Liu. **Computation with spikes in a winner-take-all network**. *Neural Computation*, **21**(9):2437–2465, 2009. 77, 97

[171] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988. 151

[172] D. Pecevski, L. Buesing, W. Maass. **Probabilistic Inference in General Graphical Models through Sampling in Stochastic Networks of Spiking Neurons**. *PLoS Comput Biol*, **7**(12), 12 2011. 74, 113, 204, 214

[173] M. Pfeiffer, B. Nessler, R. Douglas, W. Maass. **Reward-modulated Hebbian Learning of Decision Making**. *Neural Computation*, **22**:1399–1444, 2010. 6, 73, 114

[174] J.-P. Pfister, T. Toyoizumi, D. Barber, W. Gerstner. **Optimal Spike-Timing-Dependent Plasticity for Precise Action Potential Firing in Supervised Learning**. *Neural Computation*, **18**(6):1318–1348, 2006. 81

[175] J.-P. Pfister W. Gerstner. **Triplets of spikes in a model of spike timing-dependent plasticity**. *The Journal of neuroscience*, **26**(38):9673–9682, 2006. 3

[176] J. W. Pillow, J. Shlens, L. Paninski, et al. **Spatio-temporal correlations and visual signalling in a complete neuronal population**. *Nature*, **454**(7207):995–999, 2008. 162

[177] H. Poon P. Domingos. **Sum-product networks: A new deep architecture**. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 689–690. IEEE, 2011. 114

[178] M. Pospischil, Z. Piwkowska, T. Bal, A. Destexhe. **Characterizing neuronal activity by describing the membrane potential as a stochastic process**. *J Physiol Paris*, **103**(1-2):98 – 106, 2009. 195

[179] A. Pouget P. Latham. **Population Codes**. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks, 2nd ed.*, pages 893–897. MIT Press, Cambridge, MA, 2002. 51, 58

[180] M. Ranzato, F. Huang, Y.-L. Boureau, Y. Lecun. **Unsupervised learning of invariant feature hierarchies with applications to object recognition**. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR'07)*, pages 1–8, 2007. 107

[181] R. P. N. Rao. **Hierarchical Bayesian inference in networks of spiking neurons**. In *Advances in Neural Information Processing Systems*, **17**. MIT Press, 2005. 117

[182] R. P. N. Rao. **Neural models of Bayesian belief propagation**. In K. Doya, S. Ishii, A. Pouget, R. P. N. Rao, editors, *Bayesian Brain.*, pages 239–267. MIT-Press, Cambridge, MA, 2007. 12, 62, 117, 152

[183] R. P. N. Rao, B. A. Olshausen, M. S. Lewicki. *Probabilistic Models of the Brain*. MIT Press, 2002. 5, 72, 113, 151

[184] R. A. Rescorla A. R. Wagner. **Classical Conditioning II**. In A. H. Black W. F. Prokasy, editors, *A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and nonreinforcement*, pages 64–99. Appleton–Century–Crofts, 1972. 21, 28, 44, 45, 55, 60

[185] J. N. Reynolds, B. I. Hyland, J. R. Wickens. **A cellular mechanism of reward-related learning**. *Nature*, **413**:67–70, 2001. 28

[186] D. J. Rezende, D. Wierstra, W. Gerstner. **Variational Learning for Recurrent Spiking Networks**. In *Proc. of NIPS 2011*, **24**, pages 136–144. MIT Press, 2012. 118, 204

[187] M. Riesenhuber T. Poggio. **Models of object recognition**. *Nature Neuroscience*, **2**:1019–1025, 1999. 51

[188] D. L. Ringach. **Spontaneous and driven cortical activity: implications for computation**. *Current Opinion in Neurobiology*, **19**:1–6, 2009. 150

[189] E. T. Rolls G. Deco. *The Noisy Brain: Stochastic Dynamics as a Principle of Brain Function*. Oxford Univ. Press, 2010. 150

[190] D. Roth. **Learning in Natural Language**. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 898–904, 1999. 12, 29, 57

[191] D. E. Rumelhart D. Zipser. **Feature discovery by competitive learning**. In D. Waltz J. A. Feldman, editors, *Connectionist Models and Their Implications: Readings from Cognitive Science*, pages 205–242. Ablex Publishing Corporation, 1988. 115

[192] U. Rutishauser R. Douglas. **State-Dependent Computation Using Coupled Recurrent Networks**. *Neural Computation*, **21**:478–509, 2009. 73, 74, 113

[193] S. Sadaghiani, G. Hesselmann, K. J. Friston, A. Kleinschmidt. **The relation of ongoing brain activity, evoked neural responses, and cognition**. *Front Syst Neurosci*, **4**:Artikel 20, 2010. 151

[194] M. Sahani P. Dayan. **Doubly Distributional Population Codes: Simultaneous Representation of Uncertainty and Multiplicity**. *Neural Comput*, **15**:2255–2279, 2003. 152

[195] Y. Sakai, H. Okamoto, T. Fukai. **Computational algorithms and neuronal network models underlying decision processes**. *Neural Networks*, **19**(8):1091–1105, 2006. 60

[196] R. Salakhutdinov G. Hinton. **Deep boltzmann machines**. In *Proceedings of the international conference on artificial intelligence and statistics*, **5**, pages 448–455. MIT Press Cambridge, MA, 2009. 114

[197] A. Sandberg, A. Lansner, K. M. Petersson, O. Ekeberg. **A Bayesian attractor network with incremental learning**. *Network: Computation in Neural Systems*, **13**:179–194, 2002. 12, 62

[198] M. Sato. **Fast learning of on-line EM algorithm**. *Rapport Technique, ATR Human Information Processing Research Laboratories*, 1999. 74, 115, 208

[199] M. Sato S. Ishii. **On-line EM Algorithm for the Normalized Gaussian Network**. *Neural Computation*, **12**:407–432, 2000. 74

[200] C. Savin, P. Joshi, J. Triesch. **Independent Component Analysis in Spiking Neurons**. *PLoS Computational Biology*, **6**(4):e1000757, 2010. 116

[201] J. Schemmel, D. Brüderle, A. Grübl, et al. **A Wafer-Scale Neuromorphic Hardware System for Large-Scale Neural Modeling**. *Proc. of ISCAS'10*, pages 1947–1950, 2010. 205

[202] J. Schemmel, D. Brüderle, K. Meier, B. Ostendorf. **Modeling synaptic plasticity within networks of highly accelerated I&F neurons**. In *International Symposium on Circuits and Systems, ISCAS 2007*, pages 3367–3370. IEEE, 2007. 115

[203] J. Schmiedt, C. Albers, K. Pawelzik. **Spike timing-dependent plasticity as dynamic filter**. In *Advances in Neural Information Processing Systems 23*, pages 2110–2118, 2010. 140

[204] W. Schultz, P. Dayan, P. Montague. **A neural substrate of prediction and reward.** *Science*, **275**:1593–9, 1997. 60

[205] S. Shinomoto, H. Kim, T. Shimokawa, et al. **Relating Neuronal Firing Patterns to Functional Differentiation of Cerebral Cortex**. *PLoS Comput Biol*, **5**(7), 07 2009. 195

[206] E. Simoncelli D. Heeger. **A model of neuronal responses in visual area MT**. *Vision Research*, **38**(5):743–761, 1998. 206

[207] P. J. Sjöström, G. G. Turrigiano, S. Nelson. **Rate, timing, and cooperativity jointly determine cortical synaptic plasticity**. *Neuron*, **32**(6):1149–1164, 2001. 80, 81, 100, 101, 104, 118, 119

[208] W. Softky C. Koch. **The highly irregular firing of cortical cells is inconsistent with temporal integration of random EPSPs.** *J. Neuroscience.*, **13**:334–350, 1993. 195

[209] S. Song L. F. Abbott. **Cortical developing and remapping through spike timing-dependent plasticity**. *Neuron*, **32**(2):339–350, 2001. 73, 102

[210] S. Song, K. D. Miller, L. F. Abbott. **Competitive Hebbian Learning Through Spike-Timing Dependent Synaptic Plasticity**. *Nature Neuroscience*, **3**:919–926, 2000. 101, 103, 115

[211] A. Steimer, W. Maass, R. Douglas. **Belief-propagation in networks of spiking neurons**. *Neural Computation*, **21**:2502–2523, 2009. 30, 152, 204

[212] M. Steyvers, M. D. Lee, E.-J. Wagenmakers. **A Bayesian analysis of human decision-making on bandit problems**. *Journal of Mathematical Psychology*, **53**(3):168–179, 2009. 5

[213] L. P. Sugrue, G. S. Corrado, W. T. Newsome. **Matching behavior and the representation of value in the parietal cortex**. *Science*, **304**:1782–1787, 2004. 19, 26, 32

[214] L. P. Sugrue, G. S. Corrado, W. T. Newsome. **Choosing the greater of two goods: Neural currencies for valuation and decision making**. *Nature Reviews Neuroscience*, **6**(5):363–375, 2005. 26, 33, 59

[215] R. Sundareswara P. R. Schrater. **Perceptual multistability predicted by search model for Bayesian decisions**. *J Vis*, **8**:1–19, 2008. 151, 152, 153, 170, 174, 176

[216] R. S. Sutton. **Gain adaptation beats least squares**. In *Proceedings of the 7th Yale Workshop on Adaptive and Learning Systems*, pages 161–166, New Haven, CT, 1992. 55, 61

[217] R. S. Sutton A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998. 28, 32, 61

[218] J. B. Tenenbaum, T. L. Griffiths, C. Kemp. **Theory-based Bayesian models of inductive learning and reasoning**. *Trends in Cognitive Sciences*, **10**(7):309–318, 2006. 151

[219] J. B. Tenenbaum, C. Kemp, T. L. Griffiths, N. D. Goodman. **How to Grow a Mind: Statistics, Structure, and Abstraction**. *Science*, **331**(6022):1279–1285, 2011. 205

[220] **The Python Language Reference**. http://docs.python.org/reference/. 190

[221] M. Toussaint. **Probabilistic inference as a model of planned behavior**. *Künstliche Intelligenz (German Artificial Intelligence Journal)*, **3**, 2009. 151

[222] M. Toussaint C. Goerick. **A Bayesian view on motor control and planning**. In O. Sigaud J. Peters, editors, *From motor to interaction learning in robots. Studies in Computational Intelligence*, pages 227–252. Springer, 2010. 151

[223] T. Toyoizumi, J.-P. Pfister, K. Aihara, W. Gerstner. **Generalized Bienenstock-Cooper-Munro rule for spiking neurons that maximizes information transmission**. *Proc. Natl. Acad. Sci. USA*, **102**:5239–5244, 2005. 81

[224] G. Turrigiano. **Too many cooks? Intrinsic and synaptic homeostatic mechanisms in cortical circuit refinement**. *Annual Rreview of Neuroscience*, **34**:89–103, 2011. 82

[225] P. J. Uhlhaas, G. Pipa, B. Lima, et al. **Neural synchrony in cortical networks: history, concept and current status**. *Front Integr Neurosci*, **3**(17), 2009. 121

[226] P. J. Uhlhaas, F. Roux, E. Rodriguez, A. Rotarska-Jagiela, W. Singer. **Neural synchrony and the development of cortical networks**. *Trends Cogn Sci*, **14**(2):72–80, 2010. 121

[227] D. Verma R. Rao. **Goal-Based Imitation as Probabilistic Inference over Graphical Models**. In *Advances in Neural Information Processing Systems 18*, pages 1393–1400. MIT Press, Cambridge, MA, 2006. 61, 62

[228] E. Vul H. Pashler. **Measuring the Crowd Within: Probabilistic representations Within individuals**. *Psychological Science*, **19**(7):645–647, 2008. 175

[229] A. Wald J. Wolfowitz. **Optimal character of the sequential probability ratio test**. *Ann. Math. Statist.*, **19**:326–339, 1948. 57

[230] X. J. Wang. **Probabilistic decision making by slow reverberation in cortical circuits**. *Neuron*, **36**:955–968, 2002. 60

[231] A. Watt N. Desai. **Homeostatic plasticity and STDP: keeping a neuron's cool in a fluctuating world**. *Frontiers in Synaptic Neuroscience*, **2**, 2010. 205, 208

[232] T. Yang M. N. Shadlen. **Probabilistic reasoning by neurons**. *Nature*, **447**:1075–1080, 2007. 5, 6, 11, 12, 21, 22, 23, 26, 30, 59, 62, 63, 64, 65, 151, 161

[233] A. Yu P. Dayan. **Expected and unexpected uncertainty: ACh and NE in the neocortex**. In *Advances in Neural Information Processing Systems 15*, pages 157–164, Cambridge, MA, 2003. MIT Press. 49, 61

[234] A. L. Yuille. **Augmented Rescorla-Wagner and Maximum Likelihood Estimation**. In *Advances in Neural Information Processing Systems 18*, pages 1561–1568, Cambridge, MA, 2006. MIT Press. 55, 69

[235] A. L. Yuille D. Geiger. **Winner-take-all networks**. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 1228–1231. MIT Press, 2003. 27

[236] R. Zemel, Q. J. M. Huys, R. Natarajan, P. Dayan. **Probabilistic computation in spiking populations**. In *Advances in Neural Information Processing Systems 17: Proceedings of the 2004 Conference*, **17**, pages 1609–1616, 2005. 117, 176

Declaration

I herewith declare that I have produced this thesis without the prohibited assistance of third parties and without making use of aids other than those specified; notions taken over directly or indirectly from other sources have been identified as such. This thesis has not previously been presented in identical or similar form to any other Austrian or foreign examination board.

Graz, 2013-12-10