CHALMERS
UNIVERSITY OF TECHNOLOGY

Bayesian inference in networks of spiking neurons

On local learning

Master's thesis in Complex Adaptive Systems

Fabian Mikulasch

Department of Space, Earth and Environment
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2019

Bayesian inference in networks of spiking neurons

On local learning

Fabian Mikulasch

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Space, Earth and Environment
In collaboration with the Priesemann-Group at MPI-DS, Göttingen
Chalmers University of Technology
Gothenburg, Sweden 2019

Bayesian inference in networks of spiking neurons
On local learning
Fabian Mikulasch

Supervisor: Viola Priesemann, Max Planck Institute for Dynamics and Self-Organization
Examiner: Kristian Lindgren, Department of Space, Earth and Environment

Department of Space, Earth and Environment
In collaboration with the Priesemann-Group at MPI-DS, Göttingen
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2019

Abstract

So far, a unifying theory for perception and learning in biological systems is missing. A promising approach is the Bayesian framework, which allows to derive neural encoding-dynamics and learning rules from first principles. However, when a population of neurons learns to encode its input a fundamental problem arises: To properly update their weights neurons need access to the momentary population-code which is not a local quantity. Up to now in the framework of Bayesian inference in spiking networks this problem has been avoided by dismissing concurrent population-codes entirely. In the broader context of efficient coding a local update rule, approximating the correct gradient, has been proposed which relies on an error-correcting balanced-state inhibition.

In this thesis this approximation is translated into the framework of Bayesian inference in spiking networks and we show that certain statistical dependencies in the input can cause it to lead to a faulty code. Therefore, a second solution is proposed which allows the neurons to learn according to the correct gradient using only local quantities. In computer simulations we show that this update rule can effectively deal with complicated statistical dependencies in the input. We also show that they can qualitatively predict response characteristics of neurons in lower visual cortex V1. Our results suggest that dendrites potentially aid learning by maintaining a representation of the coding error in their local potentials via a forced balance of inputs on a sub-cellular level. This mechanism could provide an explanation for the observed clustering of inhibitory and excitatory synaptic connections in the cortex from the perspective of neural learning.

Keywords: Balanced state, efficient coding, spiking neurons, inference, dendritic computation, synaptic clustering.

Contents

List of Figures

Abbreviations

bAP Back-propagated action potentials, page 29
DS Dendritic scaling, page 29
EM Expectation-Maximization algorithm, page 12
GLIF Generalized leaky-integrate-and-fire neuron model, page 3
PSP Post-synaptic potential, page 8
SI Somatic-inhibition network by Bourdoukan et al., page 15
STDP Spike-timing dependent plasticity, page 4
V1 Primary areal in the lower visual cortex, page 27
WTA Winner-take-all circuit, page 3

Introduction

> On a toujours cherché des explications quand c'était des représentations qu'on pouvait seulement essayer d'inventer
>
> "We have always sought explanations when it was only representations that we could seek to invent"
>
> Paul Valéry

Animals have to interact with an enormously complex environment. In order to perform meaningful actions in it, it is essential for the animal to form a simplified model of its surroundings based on the sensory input it receives. This perception however, relying on the information the brain obtains from receptor cells, is fundamentally uncertain due to the irregular and noisy nature of the world and the sensory system itself. The animal thus faces two major challenges: Firstly to compress the very high-dimensional input into symbolic representations that enable to model it using limited resources and secondly to do so while dealing effectively with the uncertainty intrinsic to this problem.

In order to solve this task most animals adapt to their environment during ontogenesis. Especially in the case of the visual system it is known that many of them specify their representations of the world based on the visual cues they observe. In experiments with cats for example it has been found that, when raised in an environment deprived of horizontal visual cues, they will be virtually unable to detect these while their accuracy for vertical cues is increased [1, 2]. These observations are in agreement with the long-standing hypothesis that animals strive to gather maximal information about their environment given the limited resources they can make use of [3]. The efficient codes resulting from this constrained optimization have been used to explain properties of the visual system on a single-cell level [4, 5]. The way these noise-resilient and efficient representations are learned by animals is therefore of great interest for neuroscience and related fields.

1.1 Bayesian inference and the brain

Quite naturally these considerations lead to view perception as a form of probabilistic inference, as it has already been suggested by Helmholtz [6]. Bayesian inference in particular manages to capture both of the aforementioned aspects. It provides a principled way to find representations for sensory data while using probability functions that can take the uncertainty of this inference into account. Strikingly it has been found in different studies that humans often perform Bayesian-optimal observations [7]. It has also been successfully used to describe learning in human behaviour [8] and therefore takes a prominent role in the study of perception.

In the Bayesian framework probability distributions are assigned to the quantities of interest and denote a belief about their current state. In the context of inference one typically distinguishes between observed variables x and hidden (or latent) variables z that *explain* them [9]. The observed quantities then could for example be the state of the sensory system of an animal and the hidden variables could be objects, like trees and other animals, that explain this state. These hidden causes thus are never observed directly but have to be *inferred* from the senses. However, they can provide a compact representation of the sensory data that allows for more efficient processing in complex environments.

Formally the goal of Bayesian inference is to find a probability function that describes the state of the hidden variables given the observation $p(z|x)$, i.e. the *posterior*. For this it is possible to employ the Bayesian formula. If one assumes a *likelihood* of the data under the causes $p(x|z)$ (i.e. the hidden causes *generate* the data) and a *prior* probability of the causes $p(z)$, then it is possible to find the posterior via the Bayesian formula $p(z|x) = \frac{p(x|z)p(z)}{p(x)}$. This inference step can be difficult to compute, but once the posterior is available to the animal, it can base its decisions on different possible states of the world and their respective estimated probabilities.

1.2 Neural dynamics as sampling

How these Bayesian computations could be implemented by networks of spiking neurons is an ongoing matter of research. One suggested possibility is that networks of neurons generate population codes, meaning that a population of neurons code to represent a particular probability function in their activity (e.g. by having each neuron represent a probability 'kernel') [10]. Another perspective is to see the individual spikes of neurons as samples from a certain probability distribution [11, 12]. Here neurons are assigned a particular random variable and the (joint) probability function of these variables is represented in the population activity over time.

1.2.1 From GLIF to Bayesian inference

The basic assumption in the sampling approach is that neurons fire probabilistically. Biological neurons are indeed very noisy. They receive signals from thousands of afferent connections, underlie thermal and ionic fluctuations, transmitter emission is

unreliable, etc. which is reflected in their dynamics [13]. In this light noise has been discussed as a resource and not a burden for computation in neural networks [14].

One particularly simple model for noisy neurons is the generalized leaky-integrate-and-fire (GLIF) model. In the common LIF model neurons are leaky integrators of their input that fire deterministically once their potential reaches a threshold. When adding a white-noise term to the input these neurons will spike in an almost Poissonian manner [15]. Their probabilistic activation function has a sigmoidal shape and is given by $p(\text{spike}|u) = \frac{1}{2}\text{erfc}\left(\frac{\theta-u}{\sigma\sqrt{2\Delta t}}\right)$, where u is the activation of the neuron, θ is the threshold, Δt is the length of the considered time interval and σ determines the strength of the introduced noise. Thus the signal-to-noise ratio of the GLIF neuron directly influences how stochastic its spiking is.

With this background it has been proposed that neurons can sample from probability distributions underlying their activity [11]. In this framework neurons fire according to a similar sigmoidal (probabilistic) activation function. Under certain conditions it has been shown that they can perform probabilistic inference by conducting a Markov chain Monte Carlo scheme. Over time the neural dynamics will visit the state space of the corresponding random variables according to their probability function, e.g. the posterior distribution of hidden variables.

1.2.2 Winner-take-all circuits

Embracing satisfying connections between form and function like this Bayesian inference is a promising approach to explain the role of cortical microcircuits in the brain. One example of these microcircuits—reoccurring connectivity patterns between neurons that are assumed to obey a canonical function—which is of particular importance for the neuroscience community is the so-called winner-take-all (WTA) circuit [16]. The WTA network motif is ubiquitous in the cortex and as a model has been used in various computational theories of the brain, for example in hierarchical models of vision [17].

In WTA circuits neurons are arranged in a layer and compete for activation. In the classical picture only one neuron 'survives' which is firing and suppresses the activation of other neurons—more general models, where multiple neurons can be active at once have in turn been termed soft-WTA circuits. This competition is implemented by mutual inhibition between the neurons, either by having an additional population of inhibitory neurons that is driven by the competing neurons or direct inhibitory connections between them.

WTA circuits can effectively compute different functions, such as the nonlinear maximum function. They also prove useful in the context of Bayesian inference [18]. Here *explaining away* effects occur between the hidden variables, meaning that if one hidden variable explains the data well other causes get less likely[1]. This can directly

[1]Consider the situation that someone wrote a good grade in a test and there are two possible explanations: The test was very easy and/or the person is good at the subject. If we *know* that the test was easy and the person has a good grade, we have a lesser belief that they are good at the subject than if we only know that they have a good grade. The easiness of the test *explains away* the good grade.

be connected to inhibition in networks that perform inference: If two neurons code for related hidden variables, they will strongly inhibit each other. This mechanism provides an intuition how inhibition in WTA circuits could guide the sampling and inference process.

1.3 Learning in spiking networks

How networks of spiking neurons learn, for example in order to model their input, is another question of ongoing research. Recent efforts connect established theories of neural learning to the theory of Bayesian sampling [18, 19, 20, 21]. This connection could provide insight into the way animals adapt to their environment.

One of the central results of these studies is that the learning rules that can be derived from first principles in the sampling models resemble one of the most studied learning rules for spiking neurons: spike-timing dependent plasticity (STDP). A neuron applying STDP will strengthen the connection to a connecting (pre-synaptic) neuron if an input spike from this afferent neuron occurs immediately before its own output spike. This learning by using timings of neural firing can be understood as a powerful way to optimize the model a network builds of its input by correlating neural activity with the occurrence of specific patterns [18].

1.4 Content of the thesis

There is however a very general problem that arises during learning in soft-WTA circuits that encode sensory data. If multiple neurons code in an ensemble the representation of the data is not a local quantity but one that is generated by the whole population. Updating the neural connections in a way that respects this non-local coding is difficult to implement in a biologically plausible manner.

This problem of local learning has been avoided so far in the context of probabilistic inference—either by assuming that neurons in the population can not be simultaneously active due to the strong lateral inhibition [20], or simply using strictly hierarchical network architectures [19]. In both cases no complicated explaining-away effects occur and the learning can be easily localized. In relation to efficient coding a solution has been proposed by Bourdoukan et al. [22] that can successfully approximate the non-local update rules, while making assumptions about the way the networks encodes its input.

In this work we take a look at the problem of local learning from the perspective of Bayesian inference (Chapter 2). Specifically we will show how the solution to the problem of local learning in soft-WTA circuits proposed by Bourdoukan et al. [22] translates into a Bayesian framework. We will also propose another solution to the problem which suggests that dendritic branches could play a critical role for neural learning by maintaining a potential that locally approximates the coding error. The main difference between the two models turns out to be that our proposed learning scheme enables the neurons to adjust their encoding on a per-input level for novel

stimuli while the approximation by Bourdoukan et al. [22] can only consider the global input.

In order to show these differences in the dynamics of the resulting neural network models, experiments of varying difficulty will be conducted in computer simulations (Chapter 3). In the last of them, using a data-set of natural images, we will replicate results by Olshausen and Field [23] who manged to predict response characteristics of neurons in the visual cortex via sparse coding. Finally the limitations and implications of the experimental results and the significance of the proposed solution for local learning will be discussed.

Theory

In the following chapter the dynamics of the neural circuit model and the rules it can apply to learn will be developed. The task the network will have to solve is to represent a (possibly high-dimensional) time-dependent signal $\mathbf{x}(t)$ in its activity. From the outputs of all coding neurons $\mathbf{z}$ it should be possible to reconstruct the signal via a linear transformation D such that $\hat{\mathbf{x}} = D\mathbf{z}$ is close to $\mathbf{x}$.

In order to perform this reconstruction we assume that the signal $\mathbf{x}$ is distributed according to a *linear generative model* (Figure 2.2). In this view the activations $\mathbf{z}$ are hidden variables that explain—or generate—$\mathbf{x}$. In the first part of this chapter a network of spiking neurons is introduced that can sample from this model.

In the second part of this chapter these sampling dynamics will be used to update the network parameters, i.e. the weights between neurons. Since there is a direct relation between the network parameters and the model parameters this can be done by maximizing the model *log-likelihood* via gradient ascent. Finally the resulting learning rules are discussed in the light of biological plausibility.

2.1 Neural sampling theory

The goal of this section is to show how a population of neurons can sample from a given joint probability distribution and—subsequently—how a dependency of neural activity on their past firing can be included. For this we will rely on the neural sampling theory developed by Buesing et al. [11] and extend it in order to consider continuous variables as well.

Consider a joint probability distribution $p(\nu_1, ..., \nu_m)$ over m binary variables ν_j. Under certain assumptions on p Buesing et al. show that a network of m spiking neurons can sample from the distribution using its inherent dynamics. In this view we say that a neuron j is spiking iff the corresponding variable ν_j is 1. This allows the network to represent a sample $\nu_1, ..., \nu_m$ in its activity and iterate the sampling as a Markov chain over discrete time-steps.

In order to construct the network we introduce stochastically firing neurons which fire with a probability depending on their membrane potentials

$$p_{dyn}(\nu_j = 1) = \mathrm{sig}(u_j) \tag{2.1}$$

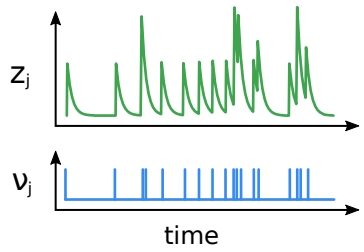

Figure 2.1: The dynamics of a single neuron over time. When it spikes ($\nu_j = 1$) the output z_j is increased by 1 immediately afterwards and subsequently decreases.

where sig is the sigmoidal function $\text{sig}(x) = [1+\exp(-x)]^{-1}$ and u_j is the membrane potential of neuron j. Here we will use a simplified version of the neural sampling theory: Two neurons are assumed never to fire at the same time, with the reasoning that in the limit of very small time-steps the probability for this event goes to zero. The neurons now will sample from p if their membrane potential satisfies the *neural computability condition*

$$u_j = \log \frac{p(\nu_j = 1 | \nu_i = 0 \text{ for } i \in \{1, ..., m\}/\{j\})}{p(\nu_j = 0 | \nu_i = 0 \text{ for } i \in \{1, ..., m\}/\{j\})}. \tag{2.2}$$

While the probability of concurrent *spike-onset* (which happens at a precise moment in time) goes to zero for small time-steps, the simultaneous *coding-activity* of neurons should be possible. We model this by introducing continuous variables z_j which model their output, or more precisely the form of the post-synaptic potentials (PSP's) they elicit in connecting neurons. Every time a neuron spikes it will cause an increase in the PSPs which exponentially decay with a time-constant τ (Figure 2.1)

$$\dot{z}_j(t) = -\tau z_j(t) + \delta(\nu_j). \tag{2.3}$$

The immediate model distribution p will depend on this past activity. This can be seen as a coarse-graining in time of the neural spiking dynamics[1]. Notably these dynamics are also consistent with viewing every neuron as a GLIF neuron with the spikes $\boldsymbol{\nu}$ as input.

Additionally we can introduce continuous inputs $\mathbf{x}(t)$, not depending on the firing, which affect the neuronal dynamics. These two additional dependencies change the joint distribution to $p(\nu_1, ..., \nu_m | \mathbf{x}, \mathbf{z})$. Finally we write the spiking probability as

$$p_{dyn}(\nu_j = 1 | \mathbf{x}, \mathbf{z}) = \text{sig}(u_j). \tag{2.4}$$

2.2 Optimal neural network

The task we want the network to solve is to represent a continuous input $\mathbf{x}$ with a given distribution $p^*(\mathbf{x})$ in its total activity $\boldsymbol{\nu} + \mathbf{z}$. In this section a suitable generative model will be introduced that defines the relation between network activity

[1] This implies that in the generative model introduced later $\boldsymbol{\nu}$ and $\mathbf{z}$ have to have the same relation to $\mathbf{x}$ in order for it to be meaningful.

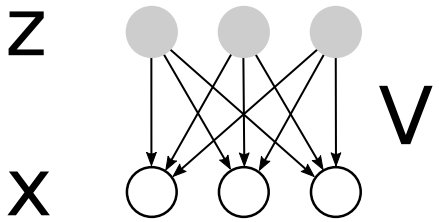

Figure 2.2: Linear generative model. When fixed, the $\mathbf{z}$ 'generate' the $\mathbf{x}$ via a linear transformation V.

and modeled input. From this we can derive the neuronal dynamics how they optimally would be in order to represent the model. In the last step, update rules for the network parameters will be derived which can be employed by the neurons to optimize the network parameters (and with that the model parameters) in order to model the input distribution.

2.2.1 Linear generative model

A generative model is defined by a joint probability distribution, in this case $p_\theta(\mathbf{x}, \boldsymbol{\nu}|\mathbf{z})$ where θ are the model parameters. This distribution contains the relation between the firing $\boldsymbol{\nu}$, neuronal output $\mathbf{z}$, and input $\mathbf{x}$. When marginalizing out the spiking $\boldsymbol{\nu}$ of the neurons we get a model for the input distribution $p^*(\mathbf{x}|\mathbf{z})$. The filtered past activity $\mathbf{z}$ is only conditioned upon, as there is no interest in drawing samples from it or modeling it—what is supposed to be modeled is $\mathbf{x}$.

We separate the model into the *prior* $p_\theta(\boldsymbol{\nu}|\mathbf{z})$ on the network activity and *likelihood* $p_\theta(\mathbf{x}|\boldsymbol{\nu}, \mathbf{z})$ of the input under the activity as

$$p_\theta(\mathbf{x}, \boldsymbol{\nu}|\mathbf{z}) = p_\theta(\mathbf{x}|\boldsymbol{\nu}, \mathbf{z}) p_\theta(\boldsymbol{\nu}|\mathbf{z}). \tag{2.5}$$

For the purpose of modeling a simple network, which we will see later, we define the *likelihood* as

$$p_\theta(\mathbf{x}|\boldsymbol{\nu}, \mathbf{z}) = \mathcal{N}(x; V(\boldsymbol{\nu} + \mathbf{z}), \Sigma). \tag{2.6}$$

In words, the input $\mathbf{x}$ is Gaussian-distributed around a linear transformation of the activity with a given covariance matrix Σ (Figure 2.2). Here, for simplicity, we will assume that this matrix is a scaled identity matrix $\Sigma = \sigma^2 I$. This amounts to the idea that from the network activity $\mathbf{z}$ we can reconstruct an estimate of the input $\hat{\mathbf{x}} = V\mathbf{z}$ (so in this model $V = D$) which approximates the real input $\mathbf{x}$ with a standard deviation of σ per dimension. The $\boldsymbol{\nu}$ do not contribute to the reconstruction in continuous time and can be left away when the precise firing time is of no interest.

The *prior* is defined to be

$$p_\theta(\boldsymbol{\nu}|\mathbf{z}) = \frac{1}{Z} \exp\{\mathbf{b}^T \boldsymbol{\nu}\} \tag{2.7}$$

Since the prior factorizes all the ν_j are independent without input. Their prior probability to be 1 is therefore $p_\theta(\nu_j = 1) = \mathrm{sig}(b_j)$.

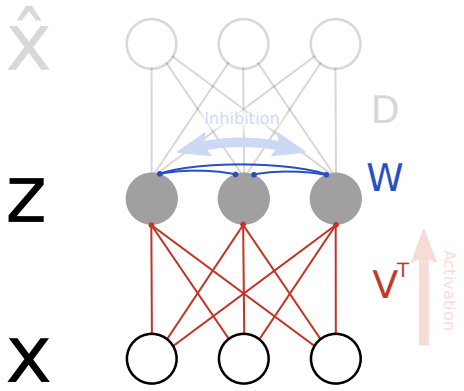

Figure 2.3: Schematic drawing of the neural network. The neurons receive feedforward input from the inputs $\mathbf{x}$ transformed by the matrix V^T and recurrent inhibition from $\mathbf{z}$ via the matrix W which in the optimal model is equal to $-V^TV$. An estimation $\hat{\mathbf{x}}$ can be decoded from the activity $\mathbf{z}$ with a matrix D (which is not part of the physical network), which in the optimal case is V. The scaling from σ^{-2} has been omitted here.

Given the *likelihood* and *prior* the *posterior* can be calculated which will be needed to derive the neural dynamics. It is given from the Bayesian formula and describes the distribution of the ν_j given the input $\mathbf{x}$ (see Appendix A.1.3.2)

$$\begin{aligned} p_\theta(\boldsymbol{\nu}|\mathbf{x},\mathbf{z}) =& \frac{p_\theta(\mathbf{x}|\boldsymbol{\nu},\mathbf{z})p_\theta(\boldsymbol{\nu}|\mathbf{z})}{p_\theta(\mathbf{x}|\mathbf{z})} \\ =& \frac{1}{Z}\exp\left(\sum_j \nu_j \left(\sum_i x_i\sigma^{-2}V_{ij} - \sum_{ik} z_k V_{ik}\sigma^{-2}V_{ij} - \frac{1}{2}\sum_i \sigma^{-2}V_{ij}^2 + b_j\right)\right) \end{aligned} \tag{2.8}$$

This calculation of the *posterior* is the *inference* step that has to be performed by the neurons. Here the normalization $Z = f(\theta, x, z)$ is difficult to calculate as we don't know $p_\theta(\mathbf{x}|\mathbf{z})$. However—as it turns out—it is not needed to be able to define the neural dynamics.

2.2.2 Neural dynamics

Using the neural sampling theory introduced in section 2.1 it is possible to define the neural dynamics that sample from the generative model defined in section 2.2.1. For that we make use of the *neural computability condition* (2.2) and the model *posterior* (2.8) which gives the membrane potentials of the neurons u_j. As the *neural computability condition* involves the fraction of the probabilities of $\nu_j = 1$ and $\nu_j = 0$, the regularization Z of equation (2.8) cancels, so it is not necessary to calculate it.

$$u_j = \underbrace{\sum_i x_i\sigma^{-2}V_{ij}}_{\text{activation}} - \underbrace{\sum_{ik} z_k V_{ik}\sigma^{-2}V_{ij}}_{\text{inhibition}} \underbrace{- \frac{1}{2}\sum_i \sigma^{-2}V_{ij}^2 + b_j}_{\text{bias}} \tag{2.9}$$

Taking a closer look at the equation for the potential u_j it is possible to separate it into three parts. Firstly the neurons receive feed-forward activation from their inputs $\mathbf{x}$, secondly recurrent inhibitory input from their own past activity and thirdly they have a constant (with respect to $\mathbf{x}$ and $\mathbf{z}$) bias term which depends on the prior and the average input estimated from V. This can be seen as corresponding to a population of inhibitory neurons forming a neural network (figure 2.3). They form synaptic connections to their inputs and—laterally—to themselves, whose scaling is determined by the model parameters θ.

Another interesting observation is that all terms except for the bias b_j are scaled with σ^{-2}, the *precision* of the model. If σ is small, i.e. the precision is large, neurons will spike less stochastically as the inputs are scaled up while the sigmoidal activation probability function stays the same. In the biological neuron this would correspond to the amount of noise in its dynamics. In the scope of this model it is possible to see it as a factor which determines how 'sure' the neurons are about their relation to the input, i.e. how well they can model it.

Finally, the spiking probability $p_{dyn}(\nu_j)$ depends on the potential u_j via equation (2.4). These two equations (2.4) and (2.9), together with the dynamics of z_j (2.3) define the complete neural dynamics. Note that in principle they can be considered without reference to the generative model defined in the last section.

2.2.3 Learning rules

When the network is sampling correctly from the generative model it is possible to use its dynamics to optimize the model parameters θ. Typically the performance of a generative model is defined to be its *log-likelihood* (see for example [9])

$$\mathcal{L}[p_\theta] = \langle \log p_\theta(\mathbf{x}) \rangle_{p^*(\mathbf{x})}. \tag{2.10}$$

Since we also introduced time-dependent variables $z(t)$ in our framework, what we actually optimize is

$$\mathcal{L}_\mathcal{T}[p_\theta] = \frac{1}{T} \int_{t=t_0}^{T} \log p_\theta(\mathbf{x}(t)|\mathbf{z}(t)). \tag{2.11}$$

For convenience however, and asserting that $\mathbf{x}(t)$ will be ergodic, we will use the former notation. Maximizing this measure with respect to θ can for example be done via gradient ascent

$$\Delta\theta \propto \frac{\partial}{\partial\theta} \mathcal{L}[p_\theta] \tag{2.12}$$

This will minimize the Kullback-Leibler divergence between the model distribution and the empirical distribution.

Unfortunately the network does not have access to the complete model distribution p_θ (specifically the posterior $p_\theta(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})$) but only to samples from it via its own dynamics $p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})$. However, it is still possible to approximately maximize the log-likelihood by maximizing a lower bound for it which is obtained by subtracting the difference of p_θ to p_{dyn} (appendix A.1.3.1)

$$\begin{aligned} \mathcal{L}[p_\theta] \geq & \langle \log p_\theta(\mathbf{x}|\mathbf{z}) - D_{KL}[p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})|p_\theta(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})] \rangle_{p^*(\mathbf{x})} \\ = & H[p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})] + \langle \log p_\theta(\boldsymbol{\nu},\mathbf{x}|\mathbf{z}) \rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})} \end{aligned} \tag{2.13}$$

For the optimization the powerful **EM**-algorithm is employed which proceeds in two steps and is guaranteed to converge. In the **E**xpectation step the model p_θ distribution is approximated by the distribution p_{dyn} (minimizing $D_{KL}[p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})|p_\theta(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})]$). This is done by drawing correct samples via the neural sampling theory. In the **M**aximization step this approximate distribution—i.e. the samples—is used to maximize the joint log-probability $\log p_\theta(\boldsymbol{\nu},\mathbf{x}|\mathbf{z})$. Because the network draws samples of $\boldsymbol{\nu}$ from p_θ online, there is no need for the whole distribution to be present in the network and there is no problem in performing the optimization.

In the following sections update rules for the feed-forward weights V, the bias $\mathbf{b}$ and the precision σ^{-2} will be derived.

2.2.3.1 Feed-forward weights

From equations (2.8), (2.12) and (2.13) it is straightforward to derive the update-rule for the weights V, which connect the neurons to the input. We perform gradient ascend on the joint log-probability, as the entropy term in (2.13) doesn't directly depend on θ

$$\begin{aligned}\Delta V_{ij} \propto & \frac{\partial}{\partial V_{ij}} \langle \log p_\theta(\boldsymbol{\nu},\mathbf{x}|\mathbf{z}) \rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})} \\ = & \langle \sigma^{-2} z_j (x_i - \sum_k V_{ik} z_k) \rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})}\end{aligned} \tag{2.14}$$

The resulting equation poses two problems. For one the sum can be identified with the estimated input $\sum_k V_{ik} z_k = \hat{x}_i$ and this estimation depends on the activity z_k of all neurons which is not a local quantity. This problem will be addressed in a later section. The second problem is the precision σ^{-2} as a factor in front. This can be solved by appealing to the theory of covariant optimization.

The problem with having the precision as prefactor is that for large precision the step-size of the updates will be large. In other words, the size of the updates depends on the steepness of the goal function, which means that simply using the gradient can lead to erratic behaviour during optimization. We can see this by noticing that the update should be in units of $[V_{ij}]$, whereas gradient ascend proposes a rule in units of $[V_{ij}]^{-1}$ which cannot always be compensated for by the learning rate. This problem can be solved by multiplying with a factor given by $[-\partial_{V_{ij}} \partial_{V_{ij}} \mathcal{L}]^{-1}$, making the algorithm covariant (see [24]). It is not possible to do this here as this would remove data-dependent terms (z_j), but the rule can still be made less dependent on the steepness by multiplying with the data independent term of the factor, which simply is σ^2. This means we can stabilize the performance of the optimization by removing the precision as prefactor.

Finally we can write the update rule in this simpler form, which makes clear that neurons should try to minimize the error of the representation when they are active

$$\Delta V_{ij} \propto \langle z_j (x_i - \hat{x}_i) \rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})} \tag{2.15}$$

Another interpretation of this is rule is as a Hebbian plasticity rule where the weight grows larger when $z_j x_i$ is correlated, but at the same time it is regularized by the correlations with the population output $\sum_k V_{ik} z_k z_j$.

2.2.3.2 Bias

The update rule for the bias can be derived analogously as

$$\begin{aligned}\Delta b_j \propto & \frac{\partial}{\partial b_j}\langle \log p_\theta(\boldsymbol{\nu}, \mathbf{x}|\mathbf{z})\rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})} \\ = & \langle \nu_j - \mathrm{sig}(b_j)\rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})}\end{aligned} \tag{2.16}$$

The resulting rule is enforcing the consistency of the model with the dynamics, i.e. the model prior $\mathrm{sig}(b_j)$ tries to match the empirical firing probability $\langle \nu_j \rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})}$.

Another way to choose the bias is to enforce a given firing rate via *homeostatic plasticity*. An approach to do that which fits well into the framework of sampling networks has been presented in detail by Habenschuss et al. [20]. Here the dynamics p_{dyn} are constrained during the **E**-step and the **EM**-algorithm will find a solution the satisfies the constrains and maximizes the likelihood. Specifically p_{dyn} is constrained to be in the set of homeostatic distributions where the empirical firing rate is equal to an homeostatic firing rate ρ_j which is chosen beforehand

$$\{p_{dyn} : \langle \nu_j \rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})} = \rho_j, \text{ for all } j = 1...m\}. \tag{2.17}$$

This constraint optimization problem can be solved with the help of Lagrange multipliers β_j. When finding p_{dyn} in the **E**-step the function to maximize is not only the negative Kullback-Leibler divergence but

$$\left\langle -D_{KL}[p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})|p_\theta(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})] + \sum_j \beta_j \left(\langle \nu_j \rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})} - \rho_j\right)\right\rangle_{p^*(\mathbf{x})} \tag{2.18}$$

This leads to the introduction of homeostatic biases $b'_j := b_j + \beta_j$ (which from now we will refer to as b_j as well for notational simplicity) and the update rule

$$\Delta b_j \propto \langle \rho_j - \nu_j \rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})} \tag{2.19}$$

2.2.3.3 Precision

For the precision

$$\begin{aligned}\Delta\sigma \propto & \frac{\partial}{\partial\sigma}\langle \log p_\theta(\boldsymbol{\nu}, \mathbf{x}|\mathbf{z})\rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})} \\ = & \left\langle \sigma^{-3}\left((\mathbf{x}-\hat{\mathbf{x}})^T(\mathbf{x}-\hat{\mathbf{x}}) - n\sigma^2\right)\right\rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})}\end{aligned} \tag{2.20}$$

where n is the number of inputs. We face the same issue with the gradient as for V and solve it in the same way, resulting in the update rule

$$\Delta\sigma \propto \left\langle \sigma^{-1}\left((\mathbf{x}-\hat{\mathbf{x}})^T(\mathbf{x}-\hat{\mathbf{x}}) - n\sigma^2\right)\right\rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})} \tag{2.21}$$

Again this rule implies consistency of the model variance with the empirical variance. If the variance is small, i.e. the neurons model their input precisely, they will fire more deterministically.

2.3 Biologically plausible learning rules

So far the derived (optimal) update rules have been introduced without discussing how they possibly could be implemented in a biological neural circuit. As mentioned before, the rules for the feed-forward weights V and the precision σ^{-2} contain the nonlocal term $\hat{x}_i = \sum_k V_{ik} z_k$ which depends on the activity z_k of all other neurons as well as the weights V_{ik} connecting them to input i. This term is needed because an optimal update the model asks for the error of the current reconstruction $\boldsymbol{\epsilon} = \mathbf{x} - \hat{\mathbf{x}}$. To implement these rules is therefore not feasible for a biological neuron that can only make use of local quantities. Another problem which hasn't been discussed is the implementation of the recurrent inhibition. From the derivation in the last section follows that the inhibitory connections should have the form $W = \sigma^{-2} V^T V$, but how this connectivity comes about is not immediately obvious.

In this section two solutions which solve both problems will be presented. In the first solution a learning rule for W is derived from a consistency condition on the network-dynamics with the generative model. Subsequently the learning rule for V is localized while making assumptions about the error of the reconstruction. In the second solution inhibitory connections to the dendrites of the other neurons are learned such that their potential is a local approximation of the error. This error can then be used to optimize the model parameters.

2.3.1 Solution 1: Learned somatic inhibition

2.3.1.1 Inhibitory weights

The main insight in this solution, which was developed by Bourdoukan et al. [22], is that the neurons receive the optimal input if they are driven by the coding error $\mathbf{x} - \hat{\mathbf{x}}$. Let's rewrite the potential of the neurons from equation (2.9) as

$$\begin{aligned} \mathbf{u} =& \sigma^{-2} \left(\underbrace{V^T \mathbf{x} - V^T V \mathbf{z}}_{V^T(\mathbf{x}-\hat{\mathbf{x}})} - \frac{1}{2} \mathrm{diag}(V^T V) \right) + \mathbf{b} \\ =& \sigma^{-2} \left(V^T \mathbf{x} + W \mathbf{z} + \frac{1}{2} \mathrm{diag}(W) \right) + \mathbf{b} \end{aligned} \tag{2.22}$$

where we contracted the inhibition into the matrix $W = -V^T V$. Realizing that $\hat{\mathbf{x}} = V\mathbf{z}$ and refactoring reveals how the sum of activation and inhibition is proportional to the coding error, which is balanced if the error is small. However, $W = -V^T V$ is only the optimal inhibition if V is correctly regularized (then $V = D$, where D are the decoding weights), otherwise—in order to have the error driving the input—the inhibition should be $W = -V^T D$. In this section we will derive update rules for the recurrent inhibition W which will force this balance. In the end this enables the network to locally learn the feed-forward weights V.

Bourdoukan et al. come up with the appropriate learning rule by noting that W should be such that $V^T \mathbf{x} + W \mathbf{z}$ is zero most of the time. Subsequently they show that the resulting rule converges to the desired solution $W = -V^T D$. We will give

a different account and derive the update rule from consistency constrains on the network dynamics.

The basic idea is that—no matter what the input is—in the long run the observed distribution of the firing $p_{dyn}(\nu_j|z,x)$ should be following the prior distribution $p_\theta(\nu_j)$ we assume it obeys. This is reasonable since in the case of optimal V this would be enforced by the inhibition $-V^TV\mathbf{z} - \frac{1}{2}\text{diag}(V^TV)$ which we replace. The inhibition then will decorrelate the coding neurons. This goal can be quantified as an optimization goal by stating that the difference between the distributions should be minimized (similar to the **E**-step)

$$\begin{aligned} 0 &\stackrel{!}{=} -\langle D_{KL}(p_{dyn}(\nu_k|\mathbf{z},\mathbf{x})|p_\theta(\nu_k))\rangle_{p^*(x)} \\ &= \left\langle \log\frac{\exp(b_k)^{\nu_k}}{1+\exp(b_k)} - \log\frac{\exp(u_k)^{\nu_k}}{1+\exp(u_k)} \right\rangle_{p_{dyn}(\nu_k|z,x)p^*(x)} \end{aligned} \tag{2.23}$$

W can then be optimized via gradient descent on the distance function (see appendix A.1.3.4). The derivation for this rule relies on the limit of $dt \to 0$.

$$\begin{aligned} \Delta W_{jk} &\propto -\frac{\partial}{\partial W_{jk}}\langle D_{KL}(p_{dyn}(\nu_j|\mathbf{z},\mathbf{x})|p_\theta(\nu_j))\rangle_{p^*(x)} \\ &\approx -\langle z_k(u_j - b_j)\rangle_{p_{dyn}(\nu_j|z,x)p^*(x)} \end{aligned} \tag{2.24}$$

Thus, whenever neuron k is active it forces neuron j via connection W_{jk} to fire according to the prior $p_\theta(\nu_j)$ which is equivalent to $u_j = b_j$. This rule is the same as derived in Bourdoukan et al. [22] and therefore has the same implications regarding the reduction of the coding error.

This solution is termed *somatic* inhibition (**SI**-network) as all weights W_{jk} need access to the same postsynaptic potential u_j. In the biological neuron this quantity would be easiest to actualize in the cell-body (soma), a place where all currents come together. Therefore, in this picture the inhibitory synapses W_{jk} connect close to the soma.

2.3.1.2 Feed-forward weights

The other problem, the non-local term in the learning rule for V, still remains. Luckily learning W with (2.24) will help to keep the coding error in check even if the feed-forward weights are not optimal, as long as they cover the data-manifold. This makes it possible to approximate the gradient ΔV. Specifically we will assume that $\mathbf{x} - \hat{\mathbf{x}} \propto \mathbf{x}$ on average. Then the problematic term $\hat{\mathbf{x}}$ can be discarded since the input $\mathbf{x}$ then is exactly what should be learned.

In principle with the proportionality assumption one could use a purely Hebbian (STDP) rule $\Delta V_{ij} \propto z_j x_i$. In this case the weights would grow without bounds though, which we can solve by introducing a normalization term. The most straightforward choice is to demand that a single coding neuron should not over-explain the input. Even if the single neuron has no access to the decoded signal $\hat{x}_i = \sum_k V_{ij} z_j$, it can compute a lower bound for it, which is its own contribution $V_{ij} z_j$.

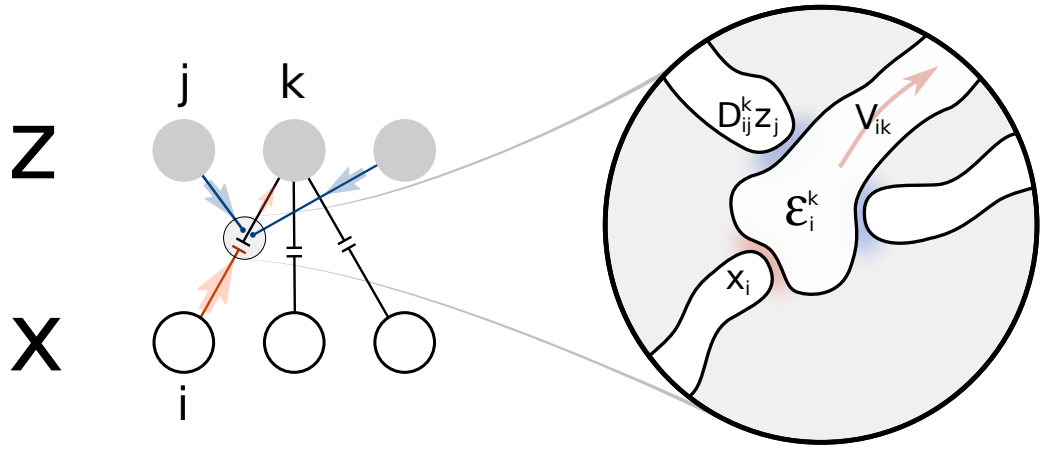

Figure 2.4: Sketch of a neural network using the proposed dendritic inhibition scheme. Here neuron i is active and propagates this activation to neuron j. Other (inhibitory) coding neurons connect with synapses to the same locus in the dendritic tree of neuron j. Via their connections D^j_{ik} they actively try to cancel the potential caused by neuron i and with this they push the local coding error potential ϵ^j_i towards zero. The remaining error potential then drives the neuron j via the weight V_{ij}.

The learning rule then reads

$$\Delta V_{ij} \propto \langle z_j(x_i - V_{ij}z_j)\rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})} \tag{2.25}$$

This results in learning an upper bound approximation for V which is proportional to the correct solution if $\mathbf{x} - \hat{\mathbf{x}} \propto \mathbf{x}$ on average.

2.3.2 Solution 2: Dendritic inhibition

Here we present a second solution to the two problems mentioned. For this we don't view the neuron as a point-neuron with u_k being the only potential located at the soma. Instead we will introduce dendritic potentials ϵ^j_i which represent the decoding error $x_i - \hat{x}_i$. In the end this solution can be seen as an argument how it could be possible for the neurons to learn with the optimal gradient on V and the correct inhibition using only local quantities.

2.3.2.1 Feed-forward weights and inhibition

Assume that on the dendrite of neuron j at the location of the x_i synapse there also are inhibitory connections $D^j_{ik}z_k$ from all coding neurons k (figure 2.4). Together they contribute to a local potential equal to the coding error

$$\epsilon^j_i = x_i - \hat{x}_i = x_i - \sum_k D^j_{ik}z_k \tag{2.26}$$

The decoding error should drive the neuron, so the neuron potential (2.9) then is

$$u_j = \sigma^{-2}\sum_i \epsilon^j_i V_{ij} - \frac{1}{2}\sum_i \sigma^{-2}V^2_{ij} + b_j \tag{2.27}$$

This is the correct error and input respectively if the weights D_{ik}^{j} are the correct decoding weights. We know however, that in the optimal case we should have $D_{ij}^{k} = V_{ij}$. Thus, if D and V are learned via the same (correct) rule the network will converge to the same solution as in the optimal case.

With this insight and ϵ_i^j defined as in equation (2.26) it is straightforward to write down the update rule for V and D using only quantities locally available in the dendrite. V_{ij} is updated according to

$$\begin{aligned} \Delta V_{ij} &\propto \langle z_j (x_i - \hat{x}_i) \rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})} \\ &= \left\langle z_j \epsilon_i^j \right\rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})} \end{aligned} \tag{2.28}$$

and D_{ij}^{k} uses the same rule including the local error-potential it connects to

$$\Delta D_{ik}^{j} \propto \left\langle z_k \epsilon_i^j \right\rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})} \tag{2.29}$$

2.3.2.2 Precision

Another advantage of maintaining a local error potential is that also the precision σ^{-2} can be learned per input. For this the covariance matrix Σ is redefined to be a diagonal matrix with entries σ_i^2 instead of using a 'global' precision. In the network the individual $\sigma_i^{j\,2} = \sigma_i^2$ then are the variances of the coding error ϵ_i^j for every j. Therefore the update rule now is

$$\Delta \sigma_i^j \propto \left\langle \sigma_i^{j\,-1} \left(\epsilon_i^{j\,2} - \sigma_i^{j\,2} \right) \right\rangle_{p_{dyn}(\boldsymbol{\nu}|\mathbf{x},\mathbf{z})p^*(\mathbf{x})} \tag{2.30}$$

Results

To evaluate the differences in the way the network using the proposed dendritic inhibition scheme, which should be equal to the 'optimal' network, and the network using somatic inhibition (**SI**-network) operate, several tests are conducted. In a first test we compare all three networks on the standard MNIST data-set of handwritten digits. For a second test we constructed an artificial data-set to specifically show the differences that can arise in the learned weights. Finally we use a complex 'real-world' data-set consisting of natural images.

In all tasks the coding neurons learn unsupervisedly to model the input distribution. They apply the learning rules while consecutively patterns are presented, for 100ms each, until convergence. During the comparisons firing rates ρ and precision σ were fixed and the performance was measured via the decoder loss $\left\langle \frac{1}{2}||\mathbf{x} - \hat{\mathbf{x}}||_2^2 \right\rangle_{p^*(x)}$ (appendix A.1.2).

3.1 Comparison on MNIST

3.1.1 Task

The MNIST data-set contains images of hand-written digits. Here—in order to keep the network small—we constrained the presented images to the digits 0, 1 and 2. The signal $\mathbf{x}$ are the pixel values of the 16×16 pixels images presented over time and faded linearly to zero between numbers. This was done in order to avoid an overlapping of the neural codes (which are coarse-grained in time) for different digits. Nine coding neurons then represent the signal in their activity (Figure 3.1).

In this test three different networks are compared. One using the optimal learning rules (optimal network, figure 3.3), one using the approximation for the update rules introduced as the first solution in section 2.3.1 (learned somatic inhibition, figure 3.3) and also an explicit implementation of the network introduced as solution 2 in section 2.3.2 (dendritic inhibition, figure A.1). The explicit implementation of the dendritic inhibition is just a proof of principle here, as we derived the dynamics specifically such that they are equal to the optimal network dynamics.

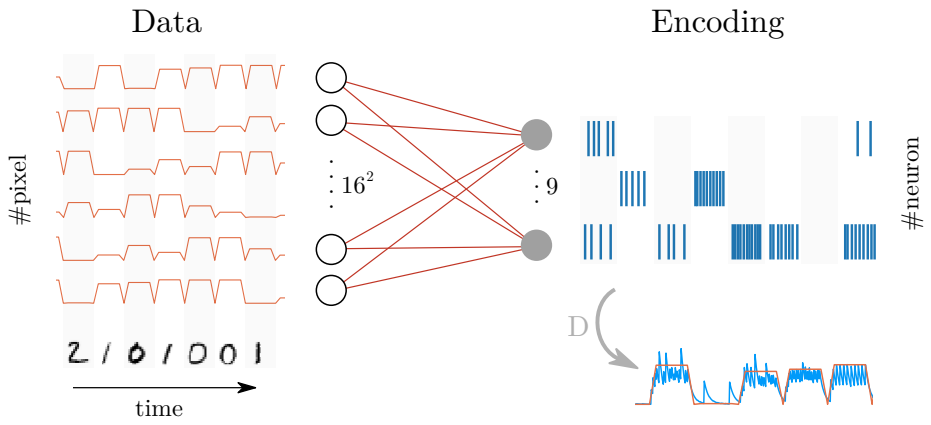

Figure 3.1: Summary of the setup of the MNIST-experiment. The digits 0, 1 and 2 are presented such that a vector of the values of the individual pixels is the input signal $\mathbf{x}$ (orange) which changes over time. The nine neurons encode the signal in their spiking dynamics which can be decoded via D to obtain the estimate $\hat{\mathbf{x}}$ (blue) which should track $\mathbf{x}$.

3.1.2 Outcome

All three networks find good codes to represent their inputs. Most of the information in the images comes from the distinction of which digit is depicted. Neurons will therefore specialize to code for either 0, 1 or 2 and also mostly code for the whole picture alone, as can be seen in the feed-forward weights. As there are more coding neurons than digits, neurons coding for the same digit will specialize to different realizations of it. If a particular stimulus lies in between those weights they will encode it together.

This occasional joint activity is reflected in the feed-forward weights in the optimal model (figure 3.3) and with that of course in the model using dendritic inhibition as well (appendix, figure A.1). The weights show a prototypical digit, but they have a 'penumbra', a negative area close to the digit where the mutual coding conflicts at times. Comparing them with the weights learned by the SI-network (Figure 3.3) we see that the latter are bigger and don't show this reflection of conflicting coding. These differences are an expected effect of the different learning rules.

All in all the resulting code for all networks is essentially the same for a simple data-set as MNIST. This is the case because here, even though the feed-forward weights are not optimal when they are learned with the approximation, the difference is small enough so that the correct inhibition can compensate for it completely. The reconstruction performance converges to the same value after learning (figure 3.2).

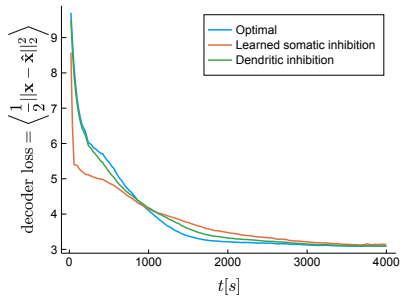

Figure 3.2: Comparison between the performances of the codes found by the different networks on MNIST. The difference in the reconstruction error in the beginning is mostly a result of the different firing rates, which haven't converged to the goal rates yet. In the end the errors converge to the same value at the same firing rate.

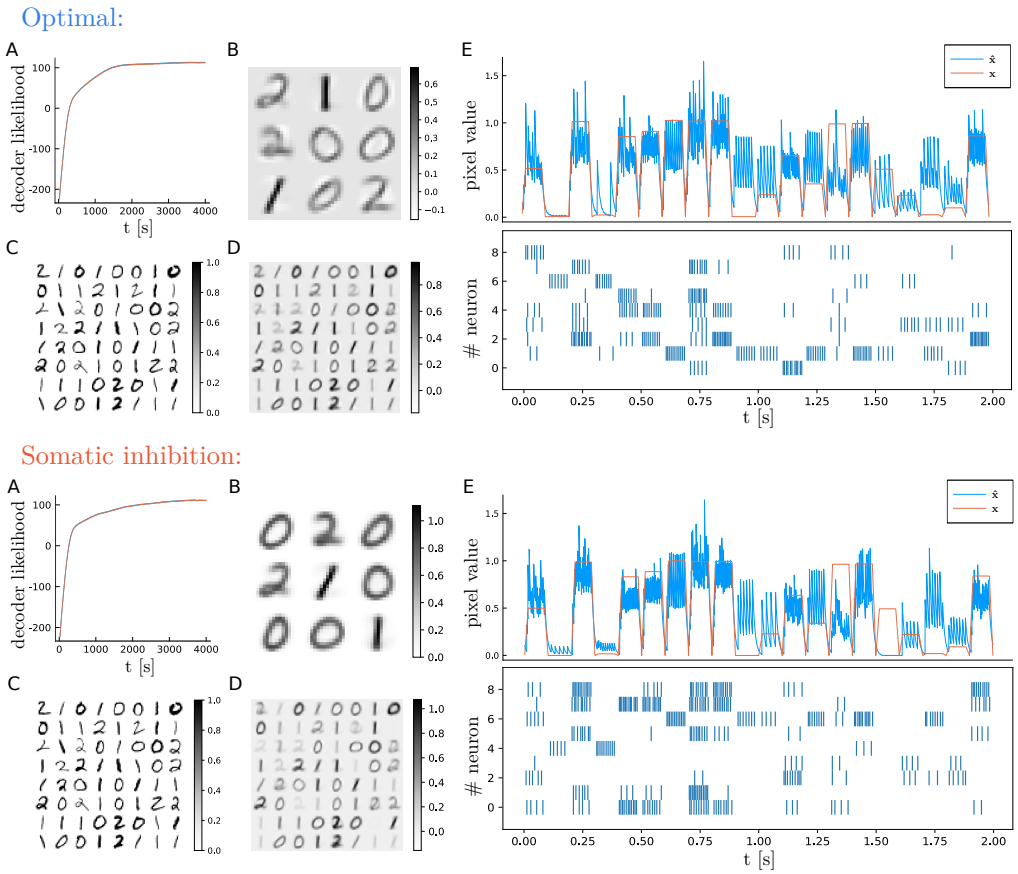

Figure 3.3: Summary of the performance of the optimal network and the SI-network in the MNIST-task. **A** Evolution of the decoder log-likelihood (i.e. the log-likelihood using the optimal decoder D and its variance) over time. **B** Resulting V-weights after learning. **C** Selection of test-inputs. **D** Reconstruction $\hat{\mathbf{x}}$ of the test-inputs averaged over the presentation time. **E** Spiking times of the network (bottom) and the comparison of inputs signal and reconstruction for a single pixel in the center of the image (top) for a set of stimuli.

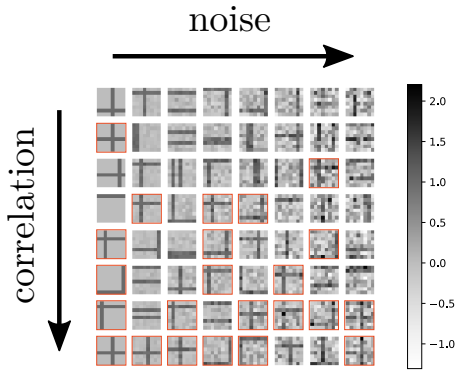

Figure 3.4: Visual guide to the correlated bars data-set. The two parameters of the data-set are the amount of white noise added onto the stimuli and the fraction of the occurrence of two bars which cross in the diagonal (red squares).

3.2 Comparison on correlated bars

In order to show clear differences in the resulting dynamics of the optimal network and the SI-network a more complicated artificial data-set was employed. The main requirement for this data is thus to have a structure such that the assumption $\mathbf{x} - \hat{\mathbf{x}} \propto \mathbf{x}$ on average does not hold. Since this assumption was made to justify the local feed-forward learning rule for the SI-network its contradiction should reflect as a decrease in performance in the encoding.

3.2.1 Task

Similarly to the MNIST-task in the last section images are presented over time and faded to zero in between the stimuli. The square images have 8^2 pixels and feature 1 or 2 of the 16 possible vertical and horizontal bars. They will be encoded by 16 coding neurons, meaning every neuron optimally should code for one of the bars since they can occur isolated.

Two additional complications are introduced (figure 3.4). One is that the images are distorted by the addition of Gaussian white noise $\boldsymbol{\xi}$ to the plain image of bars, where $\boldsymbol{\xi}$ is a Gaussian random variable with mean $\mathbf{0}$ and covariance matrix $\sigma_\xi^2 I$. The other is that certain combinations of bars can be correlated while others are always independent. For this firstly a random bar is chosen from all bars with equal probability and added to the image. Subsequently, with probability χ, a corresponding bar will be added which can be obtained by flipping the image around the diagonal (top-left to bottom-right). Otherwise, another random bar out of all 16 bars will be selected.

3.2.2 Outcome

In total four different data-sets of varying difficulty were tested. These covered the combinations bars with and without correlations ($\chi = 0.4$ and $\chi = 0.0$) and images

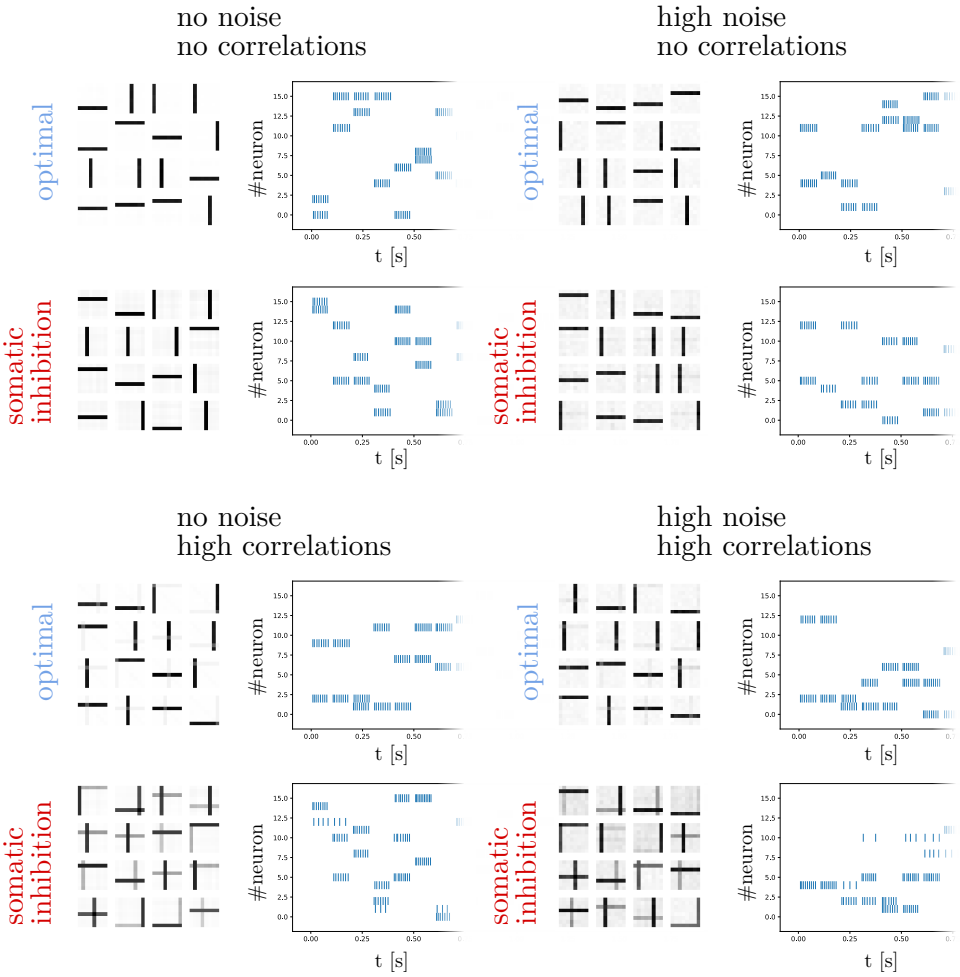

Figure 3.5: Exemplary resulting network dynamics for the correlated bars data-set. Here we compare the feed-forward weights and the spiking dynamics in response to the bar stimuli of the optimal network and the SI-network after learning. The four cases considered are data-sets with and without noise ($\sigma_\xi = 0.3$ and $\sigma_\xi = 0.0$) and with and without correlations between bars ($\chi = 0.4$ and $\chi = 0.0$).

with and without noise ($\sigma_\xi = 0.3$ and $\sigma_\xi = 0.0$). The two networks were trained with similar hyperparameters which were optimized for the task and equal for all conditions. Their resulting dynamics and feed forward weights clearly differ (Figure 3.5). While the optimal network finds relatively consistent weights and encodings for all four conditions the performance of the SI-network critically depends on the introduced correlations. Together with the goal bars their correlated corresponding bars show up in the weights as they cannot be subtracted individually. This deteriorates the code and neurons tend to spike for bars or noise they should be unspecific for.

To test this differences quantitatively 50 runs per network and condition were conducted. The neurons were trained and tested on the noisy images of correlated bars and their goal was to de-noise the images, so the decoder loss was calculated in respect to the noise-free images. In this particular task a second measure can be introduced to show how specific the firing of the coding neurons are as each neuron

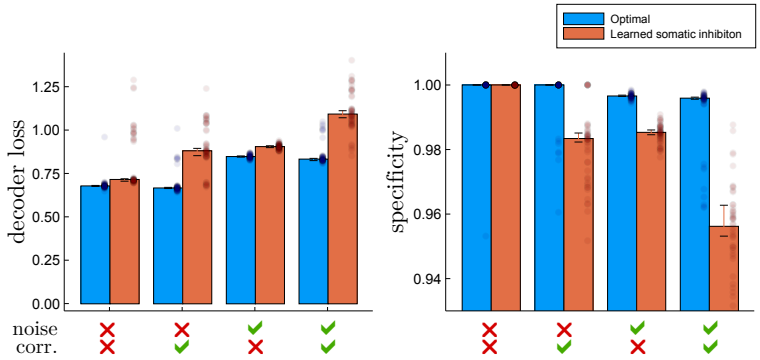

Figure 3.6: Comparison of the performances of each network in the different conditions of the bars task. Shown are the medians of the decoder loss and specificity of firing of 50 runs and the corresponding 95% bootstrapping confidence intervals.

should optimally code for exactly one bar. The specificity of the firing was defined to be the number of specific spikes (those spikes of a neuron responding to the bar it preferentially spikes for) divided by the total amount of spikes emitted.

The results are summarized in figure 3.6. In the case of highly correlated bars there clearly is a significant difference in median performance in terms of the decoder loss between the optimal and the SI-network. In fact on all conditions the optimal network performs significantly better ($p \ll 0.01$) while the SI-network happens to get stuck at a local optimum more often. The optimal network also shows a significantly better specificity ($p \ll 0.01$) except for the no noise, no correlations condition ($p = 1.0$). The difference in the decoder loss in the conditions without correlations is rather small however and the networks consistently find very similar codes whose performances to a certain extent depend on the precise hyperparameters used for learning. To summarize the experiment makes clear that—as expected—certain types of correlations in the data of an otherwise simple task can lead to a considerable decrease in performance for the SI-network while the optimal network can subtract them out.

3.3 Natural image stimuli

In a last test the optimal network was applied to a more complicated real-world data-set. Specifically we replicate the experiment performed by Olshausen and Field [23] where the sparse components of natural images predict the tuning of receptive fields of neurons in the visual cortex (V1). The same data-set of natural scenes as in the original experiments was used [25] with slightly different preprocessing. The optimal network was then applied to the images in a similar fashion as in the experiments in the last sections.

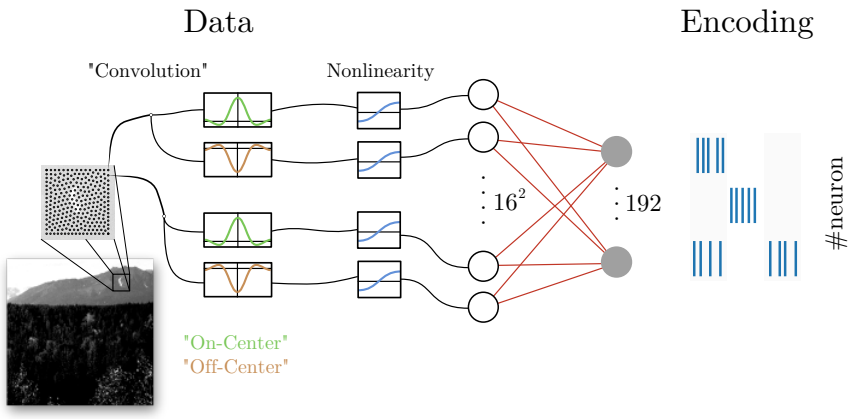

Figure 3.7: Summary of the setup of the natural scenes experiment. 16×16 patches are extracted from natural images. A linear-nonlinear model of on/off retinal ganglion cells is applied to model the input to higher visual processing areas. The output of the 16^2 'ganglion cells' is then encoded by the 192 coding neurons.

3.3.1 Task

The data-set consists of ten images of nature scenes with a resolution of 512×512 pixels. From these images random 16×16 patches are extracted. In principle these patches could be fed into the network as the images in the last experiments. However, for this approach a problem arises that originates from the fact that natural images roughly have a $1/f$ spectrum. This means that 'slow' fluctuations in the images (fluctuations of large extent) have a large effect onto the images magnitude while fast oscillations are comparatively weak. The network would thus put major emphasis on the large fluctuations, while the differences on a short length-scale mostly wouldn't be modeled. Additionally the neurons spiking is a binary process and even though they can code for continuous variables the approximately Gaussian distribution of pixel values is difficult to model with this process. The typical solution to the first problem, employed by Olshausen and Field, is to *whiten* the images, i.e. flattening the spectrum in Fourier space which amounts to convolving the images with a whitening kernel. These kernels closely resemble difference-of-Gaussians (mexican hat) functions which in turn can be used to model the receptive fields of retinal ganglion cells [26]. The array of retinal ganglion cells which preferentially react to bright spots with a dark surround or dark spots with a bright surround, termed ON- and OFF-center cells respectively, can therefore be seen as approximately whitening the incoming stimuli.

In order to provide the network with a meaningful input that it can process, the retinal ganglion cells were modeled explicitly (Figure 3.7, appendix A.2.2). First a lattice of evenly distributed 16^2 ON and 16^2 OFF-center retinal ganglion cells receives the pixel luminances as input where the receptive fields of the cells are difference-of-Gaussians functions. The cells then output this linear transformation of their input modulated by a sigmoidal activation function. This linear-nonlinear model tackles both problems mentioned by equilibrating the size of fluctuations of different length-scales and making the stimuli easier to model for spiking neurons

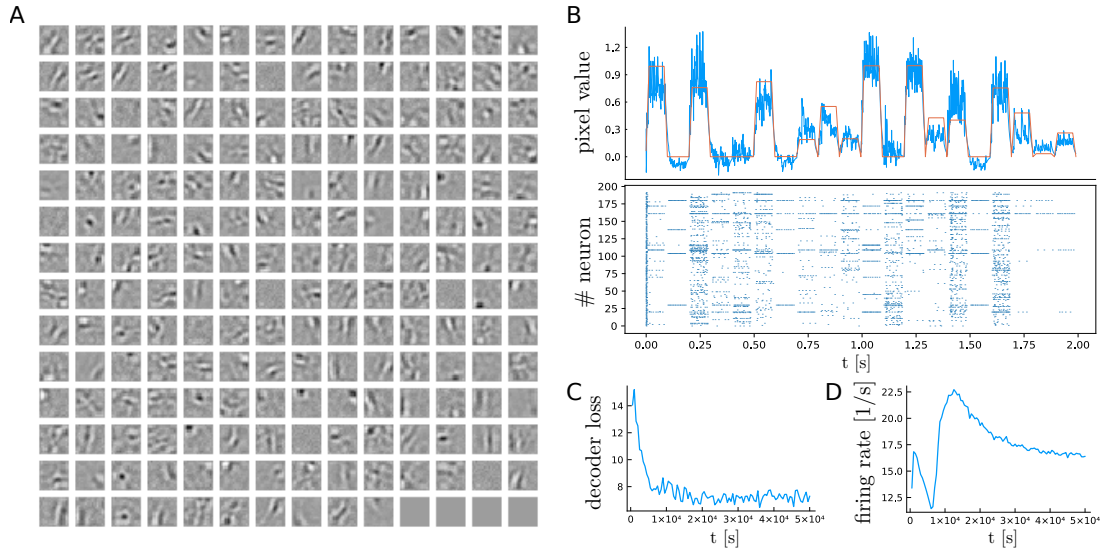

Figure 3.8: Results for the optimal network when learning natural scenes. **A** The 'receptive fields' of the 192 coding neurons obtained by stimulating them with white noise and measuring the responses. **B** Spiking times of the network (bottom) and the comparison of inputs signal and reconstruction for a single ganglion cell in the center of the image (top) for a set of stimuli. **C** Evolution of the decoder loss on a test-dataset over time. **D** Evolution of the mean firing rate per neuron over time.

by soft-thresholding them. Finally the outputs of the 2×16^2 ganglion cells were encoded by 192 coding neurons in the same way as in the tasks before with the difference that now the precisions σ_i were learned and no homeostatic constraint was put onto the firing rate.

3.3.2 Outcome

The coding neurons manage to model the input reasonably well after learning as can bee seen in the evolution of the decoder loss over time and the accuracy of $\hat{\mathbf{x}}$ tracking $\mathbf{x}$ per ganglion cell (figure 3.8). Unlike the simpler stimuli which can be modeled by few neurons the image patches are encoded by several coding neurons which often are active at the same time. Especially during the activity of a large amount of neurons the network shows an almost irregular behaviour.

Interestingly the networks performance converges quite early, after about 200,000 presented images. This is in contrast to the network parameters, which continue to change in order to optimize the code. An effect of this remodeling can be seen in the average firing rate of the neurons. It mostly decreases and converges much later, after about 500,000 presented images. The reason for this is that the log-likelihood lower bound includes an 'information cost' (see appendix A.1.3.1, the $\log p_\theta(\nu|x,z)$ term) which penalizes unnecessary (uninformative) spiking. Since a higher firing rate in principle should enable a better modelling of a continuous signal this signifies that the encoding continues to improve by using less spikes while the decoder loss keeps constant.

3.3.2.1 Receptive fields

In contrast to the tasks before the individual roles of the neurons for coding cannot be assessed by looking at the feed-forward weights. The weights only show the relations to the ganglion cells and the 'convolution' these perform on the images is not exactly invertible in this case. This prevents the possibility to infer without information loss directly from the network parameters to what stimulus (in the image domain) a neuron preferentially responds.

To be able to characterize the neural response profile in a simple way and to make them comparable to the results by Olshausen and Field a method can be employed that finds applications in experimental neuroscience: reverse correlation [27, 28]. In this approach the receptive field of a neuron is defined to be the average of the stimuli that trigger a spike. For this white noise images with a similar standard deviation as the natural images are presented to the network including the linear-nonlinear model. From the network response and the presented images the average spike triggering stimulus then can be computed. The resulting receptive fields are depicted in figure 3.8.

3.4 Discussion

There is a crucial difference between the two approaches to learning: When using point-neurons as in the approximation by Bourdoukan et al. [22], the inhibition can only act upon the projection of the error into the coding domain; the learning rule for the inhibitory weights uses the somatic potential. In contrast, when we expand the model neuron to include dendritic potentials it is possible to make use of this information which previously has been lost. It enables the neuron to tell which part of the input is not encoded well, which can have a great impact on the efficiency of learning.

The results of the experiments show that under certain conditions the approximation of Bourdoukan et al. finds a good solution. However, in general it clearly is advantageous to learn the network parameters by the proper gradient, since it allows the coding neurons to separate the high-dimensional input between them even if they are coding concurrently. This especially becomes apparent in the case of natural scenes as input. Here no convergent network with the SI approximation has been found during the work on this thesis. The results obtained in the correlated bars task therefore seem to be generalizable to other complex input statistics.

The last experiment also shows that the proposed network can qualitatively predict orientation-selective tuning of neurons in the lower visual cortex (V1). Most of them show the typical Gabor wavelet tuning (a multiplication of a 2D Gaussian and a 2D sin- or cos-function) that has been found by Olshausen and Field [23] in their experiment as well and which has been used to describe the response properties of simple cells in V1 [29]. Gabor wavelets as image filters emerge in many other computational models of vision such as slow feature analysis [30], deep learning [31], feed-forward and recurrent models [32, 33] and others. The unifying property of these different algorithms is that they find highly independent (linear) features

which carry the majority of the information that can be modeled in the images. That these components are recovered by a spiking network preforming Bayesian inference is therefore reassuring, since the generative model strives to capture precisely this underlying structure of the images in its hidden variables.

3.4.1 Plausibility of the dendritic inhibition scheme

This qualitative correspondence between the codes of the model and biological neurons is enticing, however it poses the valid question if cortical circuits really could specialize through similar mechanisms. The proposed dendritic inhibition scheme bears several idiosyncrasies that appear artificial and hard to implement in a biological system. Firstly it constrains the coding neurons to be inhibitory neurons by Dale's law, which poses that neurons can only be either inhibitory or excitatory. This could be undesirable if the encoding of the input is going to be processed further. Secondly the connection scheme demands that *every* coding neuron forms inhibitory connections to *every* dendritic subbranch of all coding neurons including its own. This would require a enormously sophisticated growing mechanism to find all the targets. It would also imply that the number of synaptic connections grows to the square of the number of coding neurons times the number of inputs which very soon would result in an unfeasible architecture.

A convenient aspect of the inhibition scheme is that a big part of these connections in most cases won't be needed. As soon as one of two neurons, which engage in mutual inhibition, is unspecific for a certain input, i.e. there are no correlations between their activity, the inhibitory connections can be omitted as their contribution would be close to zero. From a biological perspective the loss of a large portion of synaptic connections during neurogenesis in childhood is a well-known phenomenon, also known as synaptic pruning. It has been suggested that the growth and subsequent elimination of neural connections is an important factor of the brain developing for efficient information storage [34].

The number of connections can be further decreased by not only leaving away uninformative communication but also clustering communication that carries the same information. If some quantities are sufficiently correlated, e.g. the activity of a subset of coding neurons with particular inputs, then compressing this activity into a lower-dimensional representation would be a possible strategy to merge inhibitory connections. A straight-forward approach that also tackles the first problem mentioned would be to introduce a population of inhibitory neurons that receive activation by the coding neurons and mediate between them. The important aspect is that the activity of the inputs can be (linearly) decoded on a per-input level from the inhibitory population sufficiently well. If this is guaranteed then the coding neurons can make use of the same benefits for learning as before while the number of connections is minimized.

In turn, a good argument for the proposed scheme stems from studies on the locations of synaptic connections on dendrites. Several physiological studies observed the clustering of synaptic connections; most notably a recent study found that the clustering of inhibitory and excitatory synapses in the adult neocortex is governed by

learning dynamics [35]. In this light a large amount of studies have introduced dendrites as complex computational elements of neurons with the clustering of synapses being a key component (see [36] for a review).

These studies mostly stress the boost in the computational capacity of the neuron by using dendrites as computational compartments whereas here we emphasize a possible benefit for learning. Related to our suggestion a study by Maass and Legenstein [37] introduces a plasticity mechanism that creates a competition between dendritic branches. While this work considers nonlinear dendrites, interestingly two mechanisms play a central role which can be found in our model as well: dendritic scaling (DS) and back-propagated action potentials (bAP). DS can be found in our model in the learning rules for the feed-forward weights V (2.28), where the impact of the dendritic error potential ϵ_i^j onto the membrane potential u_j of the neuron is scaled instead of the individual synaptic connections. *In vivo*, a resemblant mechanism for the adaption of the coupling strength between dendritic branches and the soma has been observed in the rat hippocampus [38]. This branch strength could not only allow the dendritic potential to travel to the soma but also let emitted action potentials pass backwards to the connecting synapses. The impact of these bAP's onto local error potential would then replace the inhibitory self-connections D_{ij}^j with the branch strength V_{ij}.

Another connection to ongoing studies in neuroscience is the dependence of the learning rules on the local dendritic potential. In the derived rules the inhibitory weights in the network change such that the net-input to the neuron is close to zero. This detailed balance of excitation and inhibition on a sub-cellular level has been proposed to enable precise gating of information-flows in networks [39]. On a network level it is known that such a balanced state results in chaotic dynamics of the spiking neurons [40]. From an information theoretic perspective in this case the individual spiking is very informative which helps to render the code efficient. Therefore a tight balance between inhibition and excitation has been discussed as a simple mechanism for constructing optimal population codes [41, 22].

3.4.2 Future directions

Considering these arguments an interesting questions is how well this input-specific inhibition could be implemented while using an additional population of inhibitory interneurons. Since the encoded signal would be further compressed in this case it is not clear if the input can be reconstructed from this representation, which is necessary for the correct inhibition. In the cortex one finds approximately a portion of 20% inhibitory neurons. In this view this quota could be hypothesized to be the minimal amount of inhibitory capacity needed given the structure of the input.

Regarding the simulations in the last experiment as a biological model of the lower visual system, it is clear that a variety of important details have been left out in order to keep the model simple. The precise properties of the model of the retinal ganglion cells have been chosen ad-hoc and only with loose reference to physiological studies. Especially the outputs of the modelled ganglion cells show very different characteristics from their biological counterparts since they emit constant signals

that have no coherence over longer time-scales. While the lack of spiking behaviour can be argued to be an adequate proxy for the behaviour of retinal ganglion cells which have been observed to obey highly synchronous firing [42], the coherence of consecutive stimuli is something that was not addressed in this model.

This aspect of time dependence and correlations over time however is important for perception. Admittedly, even though spiking neurons intrinsically operate on a time dependency, the model proposed here is not able to model this additional property of the input as it only considers the momentary coding, i.e. there is no memory. It could be extended though, for example by the introduction of motion sensitive cells or by learning time dependent priors on the activity of the coding neurons. This could enable the network to model dynamic quantities such as the movement of objects or the egomotion of the observer. In any case the model bears a lot of potential for extensions in order to model additional aspects of the visual and other sensory systems.

Conclusion

Spike-based learning in graphical models is difficult to implement biologically since explaining-away effects introduce non-local dependencies in the learning rules. In this thesis we showed that in a linear model this problem can be overcome by introducing local potentials that model non-local quantities. These potentials, which represents the coding error, can be used to adjust inhibitory synapses, feed-forward impact of the inputs and even the stochasticity of the firing of the neuron. With this adaptation, derived from the generative model, the network successfully learns to model its input statistic.

In the case of natural images we could show that the response characteristics of the model neurons bear resemblance to those of neurons in V1. Furthermore, this straightforward solution to perform Bayesian computations with only local quantities predicts the clustering of inhibitory and excitatory connections on the dendrites since the inhibition has to gate already known information on a per-input level. It is remarkable that the framework of Bayesian inference can explain these properties of neural networks in the cortex—such as STDP learning rules, synaptic clustering and neural response characteristics—while relying on very few assumptions only.

In conclusion we show in an example that from the fundamental hypothesis, that animals strive to encode maximal information about their environment, optimal neural coding schemes and architectures can be derived. The potential of this approach to explain features of the nervous system is encouraging the developement of more sophisticated models that at some point might be able to capture the essence of perception and learning—from a (sub-)cellular to a behavioural level—in a unified theory.

Bibliography

[1] Colin Blakemore and Grahame F Cooper. "Development of the brain depends on the visual environment". In: *Nature* 228.5270 (1970), p. 477.

[2] Helmut VB Hirsch and DN Spinelli. "Visual experience modifies distribution of horizontally and vertically oriented receptive fields in cats". In: *Science* 168.3933 (1970), pp. 869–871.

[3] Horace B Barlow et al. "Possible principles underlying the transformation of sensory messages". In: *Sensory communication* 1 (1961), pp. 217–234.

[4] Joseph J Atick and A Norman Redlich. "Towards a theory of early visual processing". In: *Neural Computation* 2.3 (1990), pp. 308–320.

[5] Bruno A Olshausen and David J Field. "Sparse coding with an overcomplete basis set: A strategy employed by V1?" In: *Vision research* 37.23 (1997), pp. 3311–3325.

[6] Hermann Von Helmholtz. *Handbuch der physiologischen Optik*. Vol. 9. Voss, 1867.

[7] David C Knill and Alexandre Pouget. "The Bayesian brain: the role of uncertainty in neural coding and computation". In: *TRENDS in Neurosciences* 27.12 (2004), pp. 712–719.

[8] Konrad P Körding and Daniel M Wolpert. "Bayesian integration in sensorimotor learning". In: *Nature* 427.6971 (2004), p. 244.

[9] Ian Goodfellow et al. *Deep learning*. Vol. 1. MIT press Cambridge, 2016.

[10] Richard S Zemel, Peter Dayan, and Alexandre Pouget. "Probabilistic interpretation of population codes". In: *Neural computation* 10.2 (1998), pp. 403–430.

[11] Lars Buesing et al. "Neural dynamics as sampling: a model for stochastic computation in recurrent networks of spiking neurons". In: *PLoS computational biology* 7.11 (2011), e1002211.

[12] Sophie Deneve. "Bayesian spiking neurons I: inference". In: *Neural computation* 20.1 (2008), pp. 91–117.

[13] A Aldo Faisal, Luc PJ Selen, and Daniel M Wolpert. "Noise in the nervous system". In: *Nature reviews neuroscience* 9.4 (2008), p. 292.

[14] Wolfgang Maass. "Noise as a resource for computation and learning in networks of spiking neurons". In: *Proceedings of the IEEE* 102.5 (2014), pp. 860–880.

[15] Charles F Stevens and Anthony M Zador. "When is an integrate-and-fire neuron like a poisson neuron?" In: *Advances in neural information processing systems*. 1996, pp. 103–109.

[16] Rodney J Douglas and Kevan AC Martin. "Neuronal circuits of the neocortex". In: *Annu. Rev. Neurosci.* 27 (2004), pp. 419–451.

[17] Maximilian Riesenhuber and Tomaso Poggio. "Hierarchical models of object recognition in cortex". In: *Nature neuroscience* 2.11 (1999), p. 1019.

[18] Bernhard Nessler et al. "Bayesian Computation Emerges in Generic Cortical Microcircuits through Spike-Timing-Dependent Plasticity". In: *PLOS Computational Biology* 9.4 (2013), pp. 1–30. DOI: 10.1371/journal.pcbi.1003037. URL: https://doi.org/10.1371/journal.pcbi.1003037.

[19] Sophie Deneve. "Bayesian Spiking Neurons II: Learning". In: *Neural Computation* 20.1 (Jan. 2008), pp. 118–145. ISSN: 0899-7667, 1530-888X. DOI: 10.1162/neco.2008.20.1.118. URL: http://www.mitpressjournals.org/doi/10.1162/neco.2008.20.1.118 (visited on 04/03/2019).

[20] Stefan Habenschuss, Johannes Bill, and Bernhard Nessler. "Homeostatic plasticity in Bayesian spiking networks as Expectation Maximization with posterior constraints". In: *Advances in Neural Information Processing Systems*. 2012, pp. 773–781.

[21] Johannes Bill et al. "Distributed bayesian computation and self-organized learning in sheets of spiking neurons with local lateral inhibition". In: *PloS one* 10.8 (2015), e0134356.

[22] Ralph Bourdoukan et al. "Learning optimal spike-based representations". In: *Advances in neural information processing systems*. 2012, pp. 2285–2293.

[23] Bruno A Olshausen and David J Field. "Emergence of simple-cell receptive field properties by learning a sparse code for natural images". In: *Nature* 381.6583 (1996), p. 607.

[24] David JC MacKay and David JC Mac Kay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[25] Bruno A Olshausen. *Sparse net*. URL: http://www.rctn.org/bruno/sparsenet/ (visited on 05/17/2019).

[26] David Marr and Ellen Hildreth. "Theory of edge detection". In: *Proceedings of the Royal Society of London. Series B. Biological Sciences* 207.1167 (1980), pp. 187–217.

[27] EJ Chichilnisky. "A simple white noise analysis of neuronal light responses". In: *Network: Computation in Neural Systems* 12.2 (2001), pp. 199–213.

[28] Jeffrey P. Jones and Larry A. Palmer. "The two-dimensional spatial structure of simple receptive fields in cat striate cortex." In: *Journal of neurophysiology* 58 6 (1987), pp. 1187–211.

[29] S Marĉelja. "Mathematical description of the responses of simple cortical cells". In: *JOSA* 70.11 (1980), pp. 1297–1300.

[30] Konrad P Kording et al. "How are complex cell properties adapted to the statistics of natural stimuli?" In: *Journal of neurophysiology* 91.1 (2004), pp. 206–212.

[31] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[32] Andrew F Teich and Ning Qian. "Comparison among some models of orientation selectivity". In: *Journal of neurophysiology* 96.1 (2006), pp. 404–419.

[33] David C Somers, Sacha B Nelson, and Mriganka Sur. “An emergent model of orientation selectivity in cat visual cortical simple cells”. In: *Journal of Neuroscience* 15.8 (1995), pp. 5448–5465.

[34] Gal Chechik, Isaac Meilijson, and Eytan Ruppin. “Synaptic Pruning in Development: A Computational Account”. In: *Neural Computation* 10.7 (1998), pp. 1759–1777. DOI: 10.1162/089976698300017124. eprint: https://doi.org/10.1162/089976698300017124. URL: https://doi.org/10.1162/089976698300017124.

[35] Jerry L Chen et al. “Clustered dynamics of inhibitory synapses and dendritic spines in the adult neocortex”. In: *Neuron* 74.2 (2012), pp. 361–373.

[36] George Kastellakis et al. “Synaptic clustering within dendrites: an emerging theory of memory formation”. In: *Progress in neurobiology* 126 (2015), pp. 19–35.

[37] Robert Legenstein and Wolfgang Maass. “Branch-specific plasticity enables self-organization of nonlinear computation in single neurons”. In: *Journal of Neuroscience* 31.30 (2011), pp. 10787–10802.

[38] Attila Losonczy, Judit K Makara, and Jeffrey C Magee. “Compartmentalized dendritic plasticity and input feature storage in neurons”. In: *Nature* 452.7186 (2008), p. 436.

[39] Tim P Vogels and LF Abbott. “Gating multiple signals through detailed balance of excitation and inhibition in spiking networks”. In: *Nature neuroscience* 12.4 (2009), p. 483.

[40] Carl Van Vreeswijk and Haim Sompolinsky. “Chaos in neuronal networks with balanced excitatory and inhibitory activity”. In: *Science* 274.5293 (1996), pp. 1724–1726.

[41] Sophie Denève and Christian K Machens. “Efficient codes and balanced networks”. In: *Nature neuroscience* 19.3 (2016), p. 375.

[42] Jonathon Shlens, Fred Rieke, and EJ Chichilnisky. “Synchronized firing in the retina”. In: *Current opinion in neurobiology* 18.4 (2008), pp. 396–402.

[43] Lisa J Croner and Ehud Kaplan. “Receptive fields of P and M ganglion cells across the primate retina”. In: *Vision research* 35.1 (1995), pp. 7–24.

Appendix

A.1 Methods

A.1.1 Implementation

When implementing the network dynamics there are a few free parameters which alter the performance that have to be fixed. These are the neural time constant τ of equation (2.3) and the learning rates used in the update rules. Based on time constants of biological post synaptic potentials, which range between 10ms and 100ms, for all simulations τ was chosen to be 10ms. Learning rates were determined based on the task and can be found in the appendix.

An additional parameter δt which is not directly part of the neural dynamics comes in because they can only be efficiently implemented in discrete time. Sampling from the time-dependent Poisson distributions that arise from the spiking probabilities in continuous time is not feasible. Therefore the simulation proceeds in time-steps of length δt which is chosen depending on the task and the applied learning rules. The most important aspect to consider here is that the probability of two neurons spiking at the same time-step is negligible, otherwise one basic assumption of the model dynamics is violated.

A.1.2 Evaluation

A.1.2.1 Performance measures

The natural measure for the performance of the network would be what we are trying to optimize, the log-likelihood. However, a few problems arise with this measure. First of all the log likelihood critically depends on the precision σ^{-2} which can grow quite large for well-performing networks, leading to a much higher log-likelihood. This is problematic as the precision—in the physical network—is only a parameter and should not necessarily have direct influence on the perceived accuracy of the code it produces. Therefore a suitable measure can be obtained by leaving away the σ-dependent terms in the log likelihood, which yields simply the squared l2-norm of the error $\frac{1}{2}||\mathbf{x} - \hat{\mathbf{x}}||_2^2$.

This points to a second problem. $\hat{\mathbf{x}} = D\mathbf{z}$ is not directly known from the network parameters, since if we're not using the correct gradient on V, it does not hold anymore that $V = D$. In this case the optimal decoder D can be calculated by performing gradient descend on the l2-norm (Appendix A.1.3.5). Then one can also define a decoder-likelihood by finding the variance of the reconstruction. This makes it possible to determine the performance of the code alone, independently of how well the network approximates the corresponding linear generative model.

A.1.2.2 Fixed parameters for comparison

An important factor for the performance of the network is the firing rate of the neurons. The tracking of a continuous signal is better if the neurons are firing a lot. To eliminate this influence for the comparison of the feed-forward and inhibitory learning rules the firing rate was constrained. In every task including comparisons the neurons were forced to spike with a homeostatic rate ρ.

To render the dynamics of the different networks completely comparable when learning the weights another parameter to consider is the precision σ^{-2}. When using somatic inhibition there is no biologically plausible way to learn it properly—in the framework of Bourdoukan et al. the neurons spike deterministically, corresponding to a very large fixed precision. Because increased noisiness is beneficial in early learning when the weights are not adapted we will start with a big σ_{init} which exponentially decays to a fixed final value σ_{final}, i.e. it is 'learned' with the rule $\Delta\sigma \propto \langle\sigma^{-1}(\sigma_{\text{final}} - \sigma^2)\rangle_{p^*(x)p_{dyn}(\nu|x,z)}$. The initial and final value and the learning-rate are chosen depending on the task.

A.1.3 Learning rules complete derivation

A.1.3.1 Log-likelihood

$$
\begin{aligned}
\mathcal{L} \propto & -D_{KL}[p^*(x)|p_\theta(x|z)] \\
\mathcal{L} = & \langle \log p_\theta(x|z)\rangle_{p^*(x)} \\
\leq & \langle \log p_\theta(x|z) - D_{KL}[p_{dyn}(\nu|x,z)|p_\theta(\nu|x,z)]\rangle_{p^*(x)} \\
= & \left\langle \log p_\theta(x|z) - \langle \log p_{dyn}(\nu|x,z) - \log p_\theta(\nu|x,z)\rangle_{p_{dyn}(v|x,z)} \right\rangle_{p^*(x)} \\
= & \langle -\log p_{dyn}(\nu|x,z) + \log p_\theta(\nu,x|z)\rangle_{p_{dyn}(v|x,z)p^*(x)} \\
= & H[p_{dyn}(\nu|x,z)] + \langle \log p_\theta(\nu,x|z)\rangle_{p_{dyn}(v|x,z)p^*(x)} \\
= & \langle \langle \log p_\theta(x|\nu,z)\rangle_{p_{dyn}(v|x,z)} - D_{KL}[p_{dyn}(\nu|x,z)|p_\theta(\nu|z)]\rangle_{p^*(x)} \\
\overset{\text{if } p_{dyn}\approx p_\theta}{\approx} & \langle \log p_\theta(x|\nu,z)\rangle_{p_{dyn}(v|x,z)p^*(x)} - I(N;X|Z=z)
\end{aligned}
$$

A.1.3.2 Optimal model

Likelihood:

$$p_\theta(x|\nu,z) = \mathcal{N}_x(V(\nu+z),\Sigma)$$

$$
\begin{aligned}
&= \sqrt{(2\pi)^n \det \Sigma}^{-1} \exp\left(-\frac{1}{2}(x - V(\nu + z))^T \Sigma^{-1}(x - V(\nu + z))\right) \\
&\quad \text{write: } \Sigma := \sigma^2, \text{ which is a diagonal matrix with entries } \sigma_i \\
&= \frac{1}{Z(\sigma^2)} \exp\left(-\frac{1}{2}\left(x^T \sigma^{-2} x + (z^T + \nu^T) V^T \sigma^{-2} V(z + \nu)\right) + x^T V(\nu + z)\right) \\
&= \frac{1}{Z(\sigma^2, V, x, z)} \exp\left(-\frac{1}{2}\left(\nu^T V^T \sigma^{-2} V \nu\right) - z^T V^T \sigma^{-2} V \nu + x^T V(\nu + z)\right) \\
&= \frac{1}{Z(\sigma^2, V, x, z)} \exp\left(\sum_k \nu_k \left(-\sum_{ij}\left(z_j + \frac{1}{2}\nu_j\right) \underbrace{V_{ij}\sigma_i^{-2} V_{ik}}_{W} + \sum_i x_i \sigma_i^{-2} V_{ik}\right)\right) \\
&\quad \text{assuming no coincident spikes: } \nu_i \nu_j = \delta_{ij} \\
&= \frac{1}{Z(\sigma^2, V, x, z)} \exp\left(\sum_k \nu_k \left(-\sum_{ij} z_j V_{ij} \sigma_i^{-2} V_{ik} - \frac{1}{2}\sum_i \sigma_i^{-2} V_{ij}^2 + \sum_i x_i \sigma_i^{-2} V_{ik}\right)\right)
\end{aligned}
$$

Prior:

$$
\begin{aligned}
p_\theta(\nu|z) &= \frac{1}{Z(b)} \exp\left(b^T \nu\right) \\
&= \prod_k \frac{\exp(\nu_k b_k)}{1 + \exp(b_k)}
\end{aligned}
$$

Joint distribution:

$$
p_\theta(x, \nu|z) = \frac{1}{Z(\sigma^2, b, V, x, z)} \prod_k \exp\left(\nu_k \left(-\sum_{ij} z_j V_{ij} \sigma_i^{-2} V_{ik} - \frac{1}{2}\sum_i \sigma_i^{-2} V_{ik}^2 + \sum_i x_i \sigma_i^{-2} V_{ik} + b_k\right)\right)
$$

Posterior:

$$
p_\theta(\nu|x, z) \propto p_\theta(x, \nu|z)
$$

A.1.3.3 Learning rules

Feed forward:

$$
\begin{aligned}
\Delta V_{ij} &\propto \frac{\partial}{\partial V_{ij}} \langle \log p_\theta(x, \nu|z) \rangle_{p^*(x) p_{dyn}(\nu|x,z)} \\
&= \left\langle \sigma^{-2} z_j (x_i - \sum_k V_{ik} z_k) \right\rangle_{p^*(x) p_{dyn}(\nu|x,z)} \\
&\approx \left\langle \sigma^{-2} z_j (x_i - V_{ij} z_j) \right\rangle_{p^*(x) p_{dyn}(\nu|x,z)} \\
&\quad \text{if x is assumed constant the rule can be integrated} \\
&= \left\langle \sigma^{-2} \nu_j \int_{t_0}^{\infty} dt\, z_j(t) \left(x_i - V_{ij} z_j(t)\right) \right\rangle_{p^*(x) p_{dyn}(\nu|x,z)}
\end{aligned}
$$

$$
\begin{aligned}
&= \left\langle \sigma^{-2} \nu_j \int_{t_0}^{\infty} dt\, z_j(t_0) e^{-\frac{t}{\tau}} \left(x_i - V_{ij} z_j(t_0) e^{-\frac{t}{\tau}} \right) \right\rangle_{p^*(x) p_{dyn}(\nu|x,z)} \\
&= \left\langle \sigma^{-2} \nu_j \tau z_j(t_0) \left(x_i - \frac{1}{2} z_j(t_0) V_{ij} \right) \right\rangle_{p^*(x) p_{dyn}(\nu|x,z)}
\end{aligned}
$$

Bias:

$$
\begin{aligned}
\Delta b_j \propto & \frac{\partial}{\partial b_j} \langle \log p_\theta(\nu|z) \rangle_{p^*(x) p_{dyn}(\nu|x,z)} \\
= & \frac{\partial}{\partial b_j} \left\langle \sum_k \log \frac{\exp(\nu_k b_k)}{1 + \exp(b_k)} \right\rangle_{p^*(x) p_{dyn}(\nu|x,z)} \\
= & \langle \nu_j - \mathrm{sig}(b_j) \rangle_{p^*(x) p_{dyn}(\nu|x,z)}
\end{aligned}
$$

Precision:

$$
\begin{aligned}
\Delta \sigma_i \propto & \frac{\partial}{\partial \sigma_i} \langle \log p_\theta(\nu|x,z) \rangle_{p^*(x) p_{dyn}(\nu|x,z)} \\
= & \frac{\partial}{\partial \sigma_i} \left\langle -\log\left(Z(\sigma)\right) - \frac{1}{2}(x - Vz)^T \sigma^{-2} (x - Vz) \right\rangle_{p^*(x) p_{dyn}(\nu|x,z)} \\
= & \frac{\partial}{\partial \sigma_i} \left\langle -\log\left(\sqrt{(2\pi)^n \prod_j \sigma_j^2} \right) - \frac{1}{2} \epsilon^T \sigma^{-2} \epsilon \right\rangle_{p^*(x) p_{dyn}(\nu|x,z)} \\
= & \left\langle -\sigma_i^{-1} + \epsilon_i^2 \sigma_i^{-3} \right\rangle_{p^*(x) p_{dyn}(\nu|x,z)} \\
= & \left\langle \sigma_i^{-3} (\epsilon_i^2 - \sigma^2) \right\rangle_{p^*(x) p_{dyn}(\nu|x,z)}
\end{aligned}
$$

A.1.3.4 Learned inhibitory somatic inhibition

$$
\begin{aligned}
0 \stackrel{!}{=} & - \langle D_{KL}(p_{dyn}(\nu_k|z,x) | p_\theta(\nu_k)) \rangle_{p^*(x)} \\
= & \langle \log p_\theta(\nu_k) - \log p_{dyn}(\nu_k|z,x) \rangle_{p_{dyn}(\nu_k|z,x) p^*(x)} \\
= & \left\langle \log \frac{\exp(b_k)^{\nu_k}}{1 + \exp\left(b_k\right)} - \log \frac{\exp(u_k)^{\nu_k}}{1 + \exp\left(u_k\right)} \right\rangle_{p_{dyn}(\nu_k|z,x) p^*(x)} \\
& \text{for } dt \to 0, \text{ so } b_k \to -\infty \text{ and } u_k \to -\infty \\
= & \langle \exp(u_k)(b_k - u_k) + (1 - \exp(u_k))[\log(1 - \exp(b_k)) - \log(1 - \exp(u_k))] \rangle_{p^*(x)}
\end{aligned}
$$

$$
\begin{aligned}
\Delta W_{ij} \propto & - \frac{\partial}{\partial W_{ij}} \langle D_{KL}(p_{dyn}(\nu_k|z,x) | p_\theta(\nu_k)) \rangle_{p^*(x)} \\
= & \left\langle \frac{\partial u_j}{\partial W_{ij}} \exp(u_j)(b_j - u_j) - \frac{\partial u_j}{\partial W_{ij}} \exp(u_j) + \right. \\
& \left. \frac{\partial u_j}{\partial W_{ij}} \exp(u_j)(\log(1 - \exp(u_j)) + 1) \right\rangle_{p^*(x)}
\end{aligned}
$$

$$\approx \left\langle \frac{\partial u_j}{\partial W_{ij}} \exp(u_j)(b_j - u_j) \right\rangle_{p^*(x)}$$
$$\approx \langle z_i(b_j - u_j) \rangle_{p_{dyn}(\nu_k|z,x)p^*(x)}$$

A.1.3.5 Decoder

Loss:

$$\mathfrak{L} = \frac{1}{2}(x - Dz)^T(x - Dz)$$

Likelihood:

$$p_\theta(x|\nu, z) = \mathcal{N}_x(D(\nu + z), \sigma_D^2)$$

Update:

$$\Delta D \propto -\frac{\partial \mathfrak{L}}{\partial D}$$
$$= (x - Dz)z^T$$

A.2 Results

A.2.1 Additional Figures

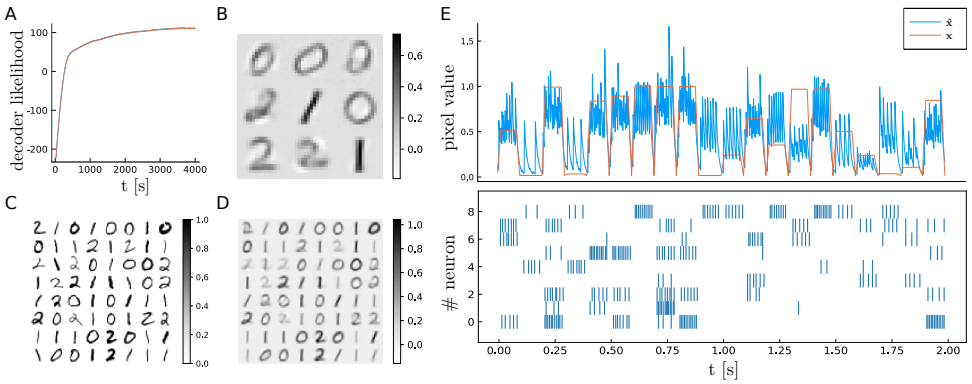

Figure A.1: Summary of the performance of the network using dendritic inhibition in the MNIST-task. Same as figure 3.3.

A.2.2 Linear-nonlinear model

16^2 points are placed randomly on a square lattice to model the centers of the retinal ganglion cells. To achieve an organic and even distribution they repel each

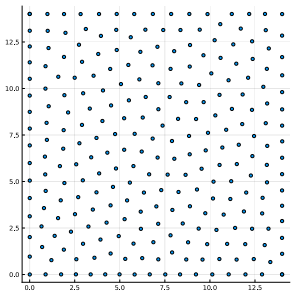

Figure A.2: Locations of the artificial retinal ganglion cells on the 16×16 pixels image.

other with forces depending on the difference of their positions $\mathbf{r}_i$: $F \propto \frac{e_{\mathbf{r}_i}}{|\mathbf{r}_i|_2^4}$. A simulation is run until convergence (figure A.2). Their receptive fields are difference-of-Gaussians functions $k_i(x, y) = \mathcal{N}(x, y; \mathbf{r}_i, \sigma_1) - \mathcal{N}(x, y; \mathbf{r}_i, \sigma_2)$ where the two 2D-Gaussians have standard deviations of $\sigma_1 = 0.7$ and $\sigma_2 = 1.12$ pixels—a ratio loosely based on physiological data [43]. These kernels $k_i(x, y)$ are positioned at the centers and multiplied with the pixel luminances $p(x, y)$ so $u_i = \sum_{x,y} k_i(x, y)p(x, y)$. The outputs of the 'cells' is calculated as $\text{out}_i^{ON} = \text{sig}(3.2u_i - 0.8)$ for the ON-cell and $\text{out}_i^{OFF} = \text{sig}(-3.2u_i - 0.8)$ for the OFF-cell.

A.2.3 Hyperparameters

Parameter	Value
dt	1ms
τ	10ms
ρ	20s^{-1}
σ_{init}	1.0
σ_{final}	0.1
η_σ	$2.0 \cdot 10^{-7}$
η_b	$7.0 \cdot 10^{-4}$
η_V	$2.0 \cdot 10^{-6}$
η_D	$2.0 \cdot 10^{-6}$

Table A.1: Figure 3.3, optimal and Figure A.1

Parameter	Value
dt	0.2ms
τ	10ms
ρ	20s^{-1}
σ_{init}	1.0
σ_{final}	0.1
η_σ	$0.2 \cdot 2.0 \cdot 10^{-7}$
η_b	$0.2 \cdot 7.0 \cdot 10^{-4}$
η_W	$0.2 \cdot 2.0 \cdot 10^{-6}$
η_V	$0.2 \cdot 2.0 \cdot 10^{-6}$
η_D	$0.2 \cdot 2.0 \cdot 10^{-6}$

Table A.2: Figure 3.3, SI

Parameter	Value
dt	1ms
τ	10ms
ρ	10s^{-1}
σ_{init}	1.0
σ_{final}	0.1
η_σ	$1.5 \cdot 10^{-6}$
η_b	$1.0 \cdot 10^{-2}$
η_V	$5.0 \cdot 10^{-5}$
η_D	$5.0 \cdot 10^{-4}$

Table A.3: Figure 3.6, optimal

Parameter	Value
dt	0.333ms
τ	10ms
ρ	10s^{-1}
σ_{init}	1.0
σ_{final}	0.1
η_σ	$1.5 \cdot 10^{-6}$
η_b	$0.333 \cdot 1.0 \cdot 10^{-2}$
η_V	$0.333 \cdot 5.0 \cdot 10^{-5}$
η_W	$0.333 \cdot 5.0 \cdot 10^{-5}$
η_D	$0.333 \cdot 5.0 \cdot 10^{-4}$

Table A.4: Figure 3.6, SI

Parameter	Value
dt	1ms
τ	10ms
σ_{init}	1.0
η_σ	$2.0 \cdot 10^{-7}$
η_b	$1.0 \cdot 10^{-5}$
η_V until $t = 3000$s	$1.0 \cdot 10^{-4}$
η_V until $t = 6000$s	$3.0 \cdot 10^{-5}$
η_V until $t = \infty$	$1.0 \cdot 10^{-5}$

Table A.5: Figure 3.8