Bayesian Methods for Neural Data Analysis

Dissertation

der Fakultät für Informations- und Kognitionswissenschaften

der Eberhard Karls Universität Tübingen

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

(Dr. rer. nat.)

vorgelegt von

Sebastian Gerwinn

aus Münster

Tübingen

2010

Tag der mündlichen Qualifikation:	22.12. 2010
Dekan:	Prof. Dr.-Ing. Oliver Kohlbacher
1. Berichterstatter:	Prof. Dr. Wolfgang Rosenstiel
2. Berichterstatter:	Prof. Dr. Matthias Bethge

Acknowledgments

First of all, I would like especially to thank my supervisor Matthias Bethge, who has always been inspirational and enthusiastic in guiding my research, as well as a good friend. I would also like to acknowledge the group of Bernhard Schölkopf as a whole; everyone has been extremely receptive, supportive and helpful to me over the course of my PhD. For many useful discussions, I would like to mention these people in particular: Peter Gehler, Matthias Hein, Frank Jäkel, Wolf Kienzle, Malte Kuss, Hannes Nickisch, Sebastian Novozin, Florian Steinke and Christian Walder. Furthermore, I would like to thank Prof. Rosenstiel for his willing review and support.

I was fortunate enough to work with Matthias Seeger, from whom I learned a lot, especially about Bayesian analysis.

I would like to thank my office mates Jakob Macke, Fabian Sinz and Ralf Häfner for many hours of stimulating whiteboard discussions, hours of entertainment and for their friendship in general. I have very much enjoyed the interaction with all members of the Computational Vision and Neuroscience Group during my time here. In particular, I would like to thank Philipp Berens, Alexander Ecker and Jan Eichorn for discussions, proofreading and again for their friendship.

My special thanks go to my parents and my sister for their caring support and patience. Finally and above all I would like to thank Laura for everything.

Contents

Abstract

Understanding the computations underlying the information processing in the nervous system is one of the major tasks in computational neuroscience. The amount of neural data is rapidly increasing. Hence, we need methods to analyze and interpret this data. Main requirements for these methods are that they can account for the variability observed in the recorded data as well as they can handle uncertainties about the underlying processing. Furthermore, they should be tractable to be applicable to large data sets. Bayesian analysis provides a principled way for incorporating these requirements as it explicitly models the involved uncertainties. In this thesis, we develop feasible Bayesian methods and apply them to simulated as well as real data. We exemplify the use of these methods on three different aspects of neural coding. First, we show how state-of-the-art models can be fitted to recorded data and obtain model based confidence intervals at the same time. Second, we show how probabilistic models can be used to extract the uncertain information about the stimulus on the basis of an observed spike train. Finally, within the framework of maximum entropy modeling, we study joint distribution of spikes and stimuli.

1

Introduction

The term 'neural code' is widely used to describe the relationship between external, sensory inputs to the brain and the internal neural response. Understanding the transformation of sensory stimuli into neural responses is a central problem in computational neuroscience. By mapping, for example, the activities of the photoreceptors in the retina into neural activities at higher areas in the visual pathway, the brain processes information and performs seemingly hard tasks such as object recognition with a remarkable precision. The purpose of this thesis is to develop and apply probabilistic methods for the analysis of neural data. By providing these tools we seek to get a better understanding of the relationship between stimuli and neural activities.

The voltage of a neuron's membrane can be observed to elicit stereotypical signals at certain points in time, which are also called action potentials or spikes (see also Figure 1.1). It is widely believed that these events are the main carrier of information between neurons. As the shape of such an action potential does not change between events, the resulting signal can also be interpreted as a sequence of discrete points in time which are called spike trains. In this thesis, we aim at predicting theses spike trains from the stimulus. With stimulus we usually mean the continuous signal arriving at the sensors. For example a visual stimulus can be parametrized by a set of pixel at a time, similar to the photoreceptors on the retina. Importantly, both signals, the neural activity as well as the stimulus can be

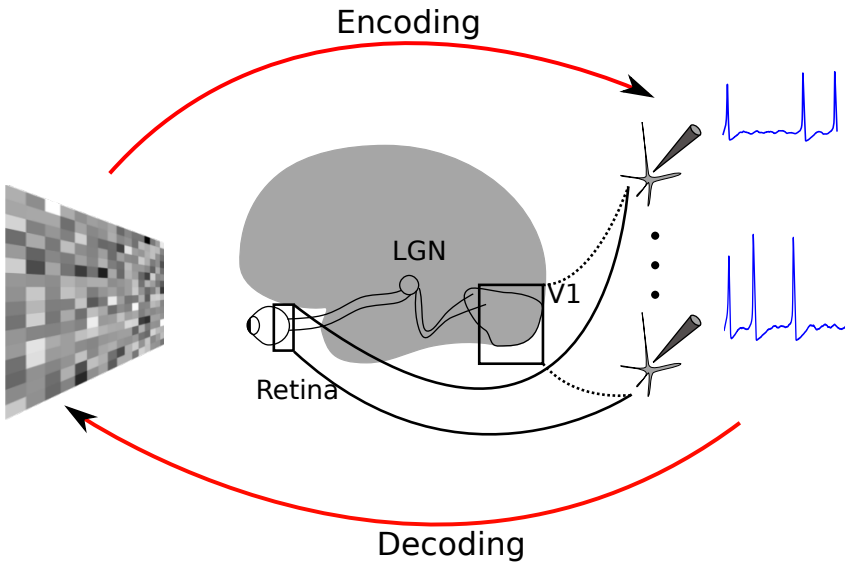

Figure 1.1: Basic setup for analyzing the coding properties of neural populations for visual input. A stimulus is presented on a screen or projected directly onto the retina. Then neural activities in form of electrical potentials are recorded. Typical locations for recording sites along the visual pathway are the retina, the lateral geniculate nucleus (LGN) and the primary visual cortex (V1). Fixing the stimulus and trying to predict the neural response corresponds to the encoding step (see also Chapter 2 and 3). Inferring the stimulus from spike trains is called decoding and is studied in Chapter 4.

considered to be generated probabilistically. When presenting the same stimulus twice the neural activity changes from trial to trial [Shadlen & Newsome 1998]. The reason can either be non-observed inputs to the neuron generated by the neural networks in the brain or intrinsic noise of the individual neuron.

For the stimulus we use the same probabilistic description as for the neural response. That is, we say a stimulus is generated according to a probability distribution $p(\mathbf{s}(t))$. For an experimental setup this distribution can freely be chosen. However, in a more realistic setting we would like to characterize all possible stimuli, which could be observed as inputs. The variety of all stimuli is huge. For example in the visual domain, there are exponentially many different combinations of pixel values. However, not all combinations are equally likely to be observed as inputs to the retina. Finding the probabilistic constraints which distinguish 'natural' from 'non-natural' images is a hard task.

When trying to characterize the relationship between stimuli and neural responses, we are interested in answering questions like:

- How is a stimulus represented or encoded in a sequence of action potentials?

- Given an arbitrary stimulus can we predict the spike-trains of ensembles of neurons?
- Can we reconstruct a stimulus which has caused a particular sequence of spikes?

The link between stimuli and responses cannot be expressed by a one-to-one mapping, that is we cannot predict the neural response exactly. However, we can assign a probability to each possible response. Mathematically speaking, we would like to have a good probabilistic description for each of the involved signals: the stimulus $\mathbf{s}(t)$, the response $\mathbf{r}(t)$ and especially the joint occurrence of a stimulus-response pair. To analyze the joint occurrence we have three possibilities. (i) We can fix a stimulus and then try to estimate how likely each of the possible responses is. (ii) We fix the response and estimate the probability of the stimulus. (iii) Finally, we can directly try to model the probability of observing a stimulus-response pair.

To estimate these probabilities, neural activity has to be measured in the response to an external stimulus. The basic experimental setup is illustrated in Figure 1.1. A stimulus is presented on a screen or directly projected onto the retina. The neural activity is then recorded in response to that stimulus. There are several methods for recording spike trains from a population of neurons. These include electrical devices such as tetrodes or multi-electrode arrays [Tolias *et al.* 2007, Zeck *et al.* 2005], but also optical devices which usually measure the change in fluorescence of certain dyes which are sensitive to the neural activity [Kerr & Denk 2008]. To build a probabilistic model as mentioned above from recorded data, we can either try to (a) estimate descriptive statistics such as the moments of the distributions or (b) fit a generative model to the observed data. The spike-triggered average for instance is a classic example for the first approach [De Boer & Kuyper 1968, Marmarelis & Naka 1972]. There, the conditional mean $\mathbb{E}[\mathbf{s}|\mathbf{r}]$ is estimated, which, in the case of white noise input can also be seen as the linear predictor for the response being a spike, given a particular stimulus. The same approach can also be applied in the decoding view leading to the optimal linear decoder [Bialek *et al.* 1991] (see also Chapter 4). Such descriptive statistics, however, cannot be used to predict spike trains. Generative models on the other hand directly model the encoding distribution $p(r|s,\theta)$, from which spike trains can be generated. Due to this ability of predicting spike trains, we will focus on generative models in this thesis.

In statistics the corresponding probabilistic object to a spike train, a set of discrete events in time, is a point process. See [Cox & Isham 1980] for a general introduction and [Daley & Vere-Jones 2005, Daley & Vere-Jones 2008] for a more

detailed analysis of the subject matter. To fit a point process to recorded data, the central quantity of interest is the likelihood of observing a spike train for a fixed set of parameters of the generative model. The likelihood in turn can be obtained for general point processes purely in terms of the conditional intensity function[1] [Barbieri *et al.* 2001, Brown *et al.* 2003]. Once the likelihood is calculated, a well known estimate for the free parameters is the maximum likelihood estimator [Pawitan 2001]. However, for classic neuron models of Hodgkin-Huxley type [Hodgkin & Huxley 1952], the computation of the conditional intensity is a non-trivial problem ([Paninski *et al.* 2004], also Chapter 2). Therefore, to fit these models to data, heuristic loss functions are often defined which measure the fitting quality of a set of parameters, see [Jolivet *et al.* 2008] and references therein. Due to the efficient inference and available performance measures, models for which a likelihood based approach is feasible became very popular for neural data analysis. A special case is the generalized linear model (GLM) for which the conditional intensity function is specified by a linear-nonlinear cascade (see Chapter 2 and Chapter 3). The first formulation of the maximum likelihood fit for multi-neural recording can be found in [Brillinger 1988, Chornoboy *et al.* 1988] and see [Boogaard 1986] for a non-neuroscience context. Of practical importance are the concavity results derived in [Paninski 2004], where conditions for the nonlinear part of the cascade are given to render the estimation problem convex. Goodness-of-fit tests are presented in [Brown *et al.* 2002]), although care has to be taken when applying those tests, see [Pillow 2009]. There are several variants of modeling the conditional intensity function. This includes the GLM with different link functions. For example [Harris *et al.* 2003] used an exponential-linear link function, instead of the widely used exponential link [Okatan *et al.* 2005, Truccolo *et al.* 2005, Pillow *et al.* 2008]. In [Weisberg & Welsh 1994] the link function is fitted jointly with the linear part in a semi-parametric fashion. In [Paninski 2003] a conditions are presented when the linear part can be recovered by spike-triggering techniques, i.e. without assuming a specific link function. Another semi-parametric approach in a non-GLM setting for analyzing the interaction between different point processes is given in [Cox 1972, Borisyuk *et al.* 1985]. There, only the modulation influence of other point processes are modeled without having to assume a specific form of the complete shape of the generating conditional intensity function. The conditional intensity function can also be fitted non-parametrically, see [Truccolo & Donoghue 2007, Coleman & Sarma 2010]. For a recent review on stat-of-the-art methods see [Brown *et al.* 2004].

Instead of classic maximum likelihood point estimation, a common theme of this

[1] The intensity function is also called hazard function in survival analysis.

thesis is to develop Bayesian inference techniques to fit generative neuron models to experimentally recorded data.

As the neural signals reflect encoding of the stimulus variables, influences of other neurons and internal dynamics, it is likely that we need complex models to explain the statistical link between stimuli and spike trains. In terms of parametric models this corresponds to a large number of parameters. Although the amount of data is increasing, the parameters are still often underconstrained by the data. That is, we have to deal with large uncertainties, not only in terms of noisy data but also in terms of uncertainty over parameters. To prevent models to fit to spurious aspects of the data — also called overfitting — an obvious way to avoid this problem is to restrict the number of parameters and hence favoring simpler models. In a Bayesian context, instead of estimating a single set of parameters a full distribution over parameters is estimated. Therefore, there is far less danger of overfitting, as the complete uncertainty over parameters is considered. Hence, there is no good reason to limit the complexity except for computationally tractability [Neal 1996]. By systematically representing the uncertainty we can make predictions based on this uncertainty which then can be tested in order to reduce the uncertainty further. We think that following a Bayesian approach can help understanding the complex computations underlying the information processing in nervous systems.

This thesis can be divided into three parts.

Part I The first part deals with modeling the encoding distribution $p(\text{spikes}|\text{stimulus})$. In Chapter 2 we introduce three popular parametric models for describing the encoding distributions which are used in this thesis. In Chapter 3 we present probabilistic methods for identifying the free parameters of the generalized linear model. In particular, we present an approximate Bayesian inference technique to estimate the uncertainty over the encoding parameters. This part is based on joint work with Jakob Macke, Matthias Seeger and Matthias Bethge and was orally presented at the Neural Information Processing System Conference 2007. It is published in two peer reviewed conference papers and in the journal 'Frontiers in Computational Neuroscience' [Seeger *et al.* 2007, Gerwinn *et al.* 2008, Gerwinn *et al.* 2010].

Part II The second part deals with deriving the decoding distribution $p(\text{stimulus}|\text{spikes})$ while fixing the encoding distribution. In Chapter 4 this is exemplified by the leaky integrate-and-fire neuron model. This part is based on joint work with Jakob Macke and Matthias Bethge and was published in 'Frontiers in Computational Neuroscience' [Gerwinn *et al.* 2009b].

Part III Finally, in Chapter 5 we present a model which characterizes the joint distribution of stimuli and spikes. The work was jointly conducted with Philipp Berens and Matthias Bethge and was published as a paper at the Neural Information Processing System conference 2009 [Gerwinn *et al.* 2009a].

2

Probabilistic models for neural populations

Empirically one finds that the neural activity varies from trial to trial, even if the stimulus is constant [Shadlen & Newsome 1998]. The underlying causes for this variability may originate either from intrinsic unreliabilities of the biophysics of the neurons, which cannot be avoided by the system or from unobserved inputs to the neuron which are difficult to control experimentally. A third possibility which has been proposed is that the variability is purposely used by the system to encode the uncertainty about the causes underlying the current stimulus [Ma *et al.* 2006]. From a modeling perspective irrespective of the source of this variability, we would like to describe the statistics of the neural activity as accurate as possible. One typically distinguishes two different classes of models: (i) mechanistic and (ii) phenomenological. The first one tries to capture specific mechanisms involved in the process of mapping the input to a neural output. The latter aims at explaining the output on a more abstract level by allowing for neglecting some of the details. There is no clear distinction between the two categories, they merely reflect different levels of abstraction. Our goal is to predict spikes as accurate as possible. Therefore, we model biophysical properties to the extend to which they are needed to improve the prediction performance. The spike generation process is usually modeled by

a two step process. The first one is a deterministic computation on the inputs, while the second is modeling random fluctuations. While we are usually interested in the deterministic computation carried out by a neuron, we need a model for the neural noise as well in order to explain the variability around the deterministic component embedded in the neural response. We follow a parametric approach for analyzing neural data, i.e. we assume a specific functional model to describe the generation of a spike in response to a presented stimulus. In this chapter we give a brief overview over the parametric models used in this thesis with the goal to illustrate the basic differences. As we are interested in modeling the neural output, we will focus on the effect of using different noise sources on the spike-generation. The models included in this chapter are the leaky integrate-and-fire model (LIF) [Tuckwell 1988, Gerstner & Kistler 2002] (see also Chapter 4), the generalized linear model (GLM) [Brillinger 1988, Paninski 2004, Okatan *et al.* 2005](Chapter 3) and the Ising model [Schneidman *et al.* 2006, Tang *et al.* 2008, Roudi *et al.* 2009c] (Chapter 5). We have ordered the models according to their degree with which they are guided by biophysical mechanisms. For instance the leaky integrate-and-fire model can be considered more mechanistic as it tries to model the membrane potential of a neuron, while the Ising model is build up by only using statistics of the neural firing.

2.1 The leaky integrate-and-fire neuron model

A popular and simple model to describe the spike generation process is the leaky integrate-and-fire neuron model [Tuckwell 1988, Gerstner & Kistler 2002]. We first describe the noiseless case consisting of the deterministic computation. Afterwards we show how neural noise can be incorporated. The noiseless model consists of a membrane potential $\mathbf{V}_t$ which accumulates the effective input $\mathbf{I}_t$. Here, $\mathbf{V}_t$ and $\mathbf{I}_t$ are scalar functions if a single neuron is modeled, or vectors if a population is considered. Whenever the membrane potential of neuron n reaches a pre-specified threshold θ^n a spike is fired and the membrane potential is reset to its resting potential, i. e. $\lim_{\varepsilon\to 0}(\mathbf{V}_{t_k+\varepsilon})_n = V_r$. In addition to the input $\mathbf{I}$, there is a leak term which drives the membrane potential back to V_r when no input is present. Correspondingly, the sub-threshold dynamics of the membrane potential can be described by the following ordinary differential equation (ODE):

$$\mathrm{d}\mathbf{V}_t = \mathbf{I}_t \mathrm{d}t - \lambda\left(\mathbf{V}_t - V_r\right)\mathrm{d}t. \tag{2.1}$$

The time constant λ specifies the time scale of the neural dynamics. Assuming the time of the last spike is t_0, the membrane potential at any time t before the next spike is given by

$$\begin{aligned}\mathbf{V}_t = &\exp\left(-\lambda\left(t-t_0\right)\right)V_r + \\ &\exp\left(-\lambda\left(t-t_0\right)\right)\int_{t_0}^{t}\exp\left(\lambda\left(s-t_0\right)\right)\mathbf{I}_s\,\mathrm{d}s =: F_{[t_0,t)}(\mathbf{I}).\end{aligned} \tag{2.2}$$

$F_{[t_0,t)}(\mathbf{I})$ is a linear functional of the stimulus $\mathbf{I}$ depending on the time of the last spike t_0 and the current time point t. Due to the additional spiking nonlinearity that governs the dynamics when the membrane potential reaches the threshold, the LIF neuron performs a complex mapping of continuous signals to spike patterns. Note that if the threshold θ of neuron is known each observed inter-spike interval defines a linear constrain on the stimulus. This is studied in more detail in chapter 4.

To include neural noise in this model, there are at least two possibilities. First, the modeled membrane potential can be subject to random fluctuations. Secondly, a possibly simpler way is to model the threshold as a random variable which is drawn every time a spike is fired. A priori it is not clear which one is better suited for modeling spike trains. Which one should be preferred, has to be decided on the basis of their prediction performance.

2.1.1 Membrane potential noise

The first possibility to model neural noise in an integrate-and-fire neuron, is to allow for fluctuations of the underlying membrane potential. The cause of this random fluctuations can be thought of as the sum of independent external random effects, or as the sum of Poisson spikes arriving at the synapse resulting in a shot-noise effect on the membrane potential, see [Burkitt 2006, Paninski *et al.* 2004]. Consequently, the ODE (2.1) turns into a stochastic differential equation (SDE):

$$\mathrm{d}V_t = \mathbf{I}_t\mathrm{d}t - \lambda(\mathbf{V}_t - V_r)\mathrm{d}t + \sigma\mathrm{d}B_t, \tag{2.3}$$

where B_t is a Brownian motion and models the random fluctuations as Gaussian[1]. The threshold θ^n for neuron n is assumed to be fixed here. If one neglects the threshold for a moment, the dynamics is purely linear resulting in a Gaussian process V_t with mean and variance given by [Oksendal & Karsten 1998, Allen 2007]:

[1]Note, that the random effects modeled by the Brownian motion are different to the ones resulting from finitely many Poisson inputs. However, due to the central limit theorem, this approximation becomes better the more Poisson inputs arriving at a synapse.

$$\begin{aligned}\mathbb{E}[V_t] &= \exp\left(-\lambda\left(t-t_0\right)\right)\left(V_r + \int_{t_0}^{t} \exp\left(\lambda\left(s-t_0\right)\right)\mathbf{I}_s\,\mathrm{d}s\right)\\ \mathrm{Var}[V_t] &= \int_{t_0}^{t} \exp\left(-\lambda(t-s)\right)^2\sigma^2\mathrm{d}s = \frac{\sigma^2}{2\lambda}\left(1-\exp\left(-2\lambda(t-t_0)\right)\right)\end{aligned} \tag{2.4}$$

We see, that the mean obeys the same dynamics as the noiseless integrate-and-fire model. Furthermore it is worth noting, that the variance of the membrane potential saturates due to the leak term. We can further generalize this model by assuming that the effective stimulus $\mathbf{I}_t$ is the result of linear filtering the actual stimulus $\mathbf{s}(t)$. That is, it is assumed that the effective stimulus $\mathbf{I}_t$ can be written as:

$$\begin{aligned}\mathbf{I}_t &= \left(\mathbf{r}\star\mathbf{s}\right)(t)\\ &= \int_{-\infty}^{\infty}\mathbf{r}(t-\tau)\mathbf{s}(\tau)\mathrm{d}\tau &(2.5)\\ &= \int_{-\infty}^{t}\mathbf{r}(t-\tau)\mathbf{s}(\tau)\mathrm{d}\tau, &(2.6)\end{aligned}$$

where $\mathbf{r}(t)$ is a linear filter acting on the recent past of the stimulus $\mathbf{s}(t)$. Equation 2.6 follows, because the linear filter is assumed to be causal, i.e. it only acts on the past of the stimulus. The filter $\mathbf{r}$ is also called a receptive field.

To illustrate the encoding, we plotted a sample spike train in Figure 2.1 (black vertical bars) together with the effective stimulus $\mathbf{I}_t$ (blue) and the corresponding membrane potential (red). The random fluctuations in the membrane potential cause non-deterministic spiking.

A problematic aspect of this neuron model is that the computation of the likelihood p(spike|stimulus) is a hard problem as the corresponding Fokker-Planck equation has to be solved, see [Paninski *et al.* 2004, Paninski *et al.* 2008]. Also it should be noted, that the model neglects some basic biophysical properties such as ion channels and spike waveforms.

2.1.2 Threshold noise

In the previous section we assumed a fixed threshold which has to be reached to fire a spikes. The membrane potential at onset of spikes, however, varies form spike to spike [Jolivet *et al.* 2006]. Therefore, the threshold is not fixed. The Gaussian noise on the membrane potential of the previous section can equivalently be seen as a continuously varying threshold. However, instead of continuously varying the threshold, we can also draw a new threshold every time after a spike has been

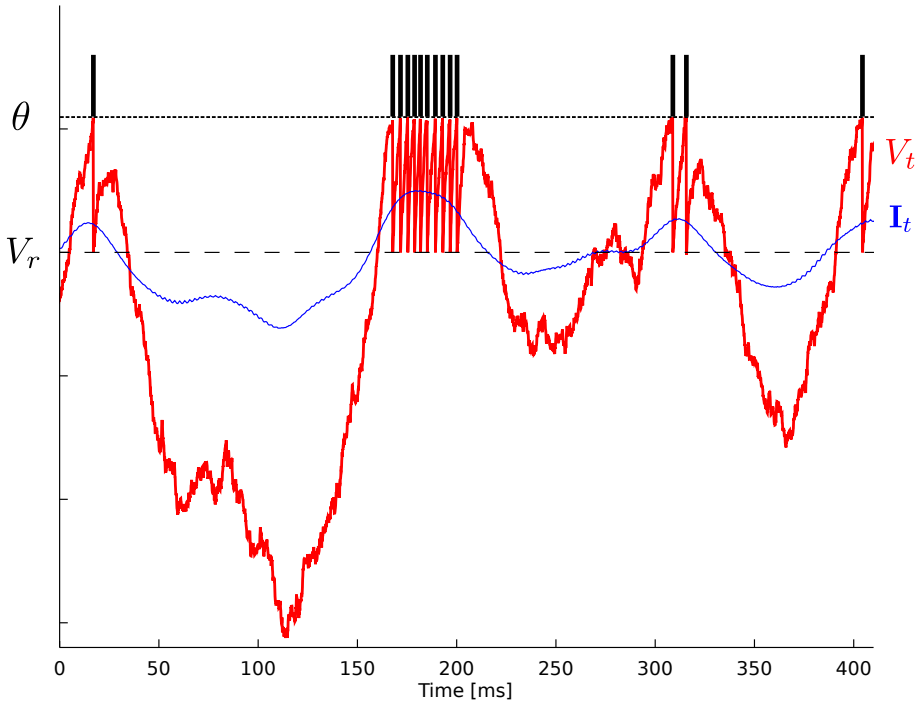

Figure 2.1: Spike train generation with a leaky integrate-and-fire neuron when membrane potential noise is used. A white noise stimulus (not shown) is filtered with a receptive field to obtain the effective input $\mathbf{I}(t)$ (blue trace). White noise is added to the effective input $\mathbf{I}(t)$ to give the membrane potential (red). Every time the membrane potential reaches the threshold θ a spike is released and the membrane potential is reset to the reset potential $\mathbf{V}_r$.

generated. As in the previous section, the time of the next spike only depends on the time of the last spike, the threshold and the time-course of the driving stimulus. Hence, conditioned on the stimulus, the resulting point process is a renewal process as the time of the next spike only depends on the timing of the very last spike. To illustrate the resulting noisy spike-generation process, we sampled a spike-train from such a neuron with every other parameter set to the same value as in the previous section. In Figure 2.2 we see, that the resulting spike-train (green) is similar to the one generated from the integrate-and-fire model with membrane potential noise (black vertical bars), however not identical. For example, we see that the first spike in the membrane-potential noise setting is missing in the threshold noise case, as the threshold happens to be too large to be reached by the (deterministic) membrane potential (plotted in red). Because of the leak term the membrane potential is driven towards zero proportional to its current value. Therefore, larger values for the membrane potential are unlikely.

2.2 The generalized linear model for spiking neurons

Compared to the two flavors of the leaky integrate-and-fire model the Generalized Linear Model (GLM) [Brillinger 1988, Paninski 2004, Okatan *et al.* 2005] is more phenomenological or abstract as it directly aims at modeling the likelihood of spikes. It has the main advantage of being computationally efficient yet flexible. In the simplest form of the GLM, spike-trains are assumed to be distributed according to an inhomogeneous Poisson process. This special case of the GLM is also known as the Linear-Nonlinear Poisson model [Simoncelli *et al.* 2004]. Specifically, the rate can be written as a Linear-Nonlinear cascade:

$$\lambda(t) = f(\mathbf{s}(t)^\top \mathbf{w}_s) \tag{2.7}$$

First, the stimulus is filtered with a parameter vector $\mathbf{w}_s$ which is referred to as the *receptive field* of the neuron. This linear filtering is similar to equation (2.6) and hence can be thought of calculating the effective input to the GLM. Subsequently, the pointwise monotonic nonlinearity f transforms the real-valued output of the linear filtering into a nonnegative instantaneous firing rate. If the current stimulus has a strong overlap with the receptive field, that is if $\mathbf{s}(t)^\top \mathbf{w}_s$ is large, this will yield a large probability of firing. If it is strongly negative, the probability of firing will be zero or close to zero. Therefore the spike generation can also be interpreted as a soft threshold integrate-and-fire model [Koyama & Paninski 2009]. However there is no reset mechanism in the GLM. The linear filtering step can also be extended

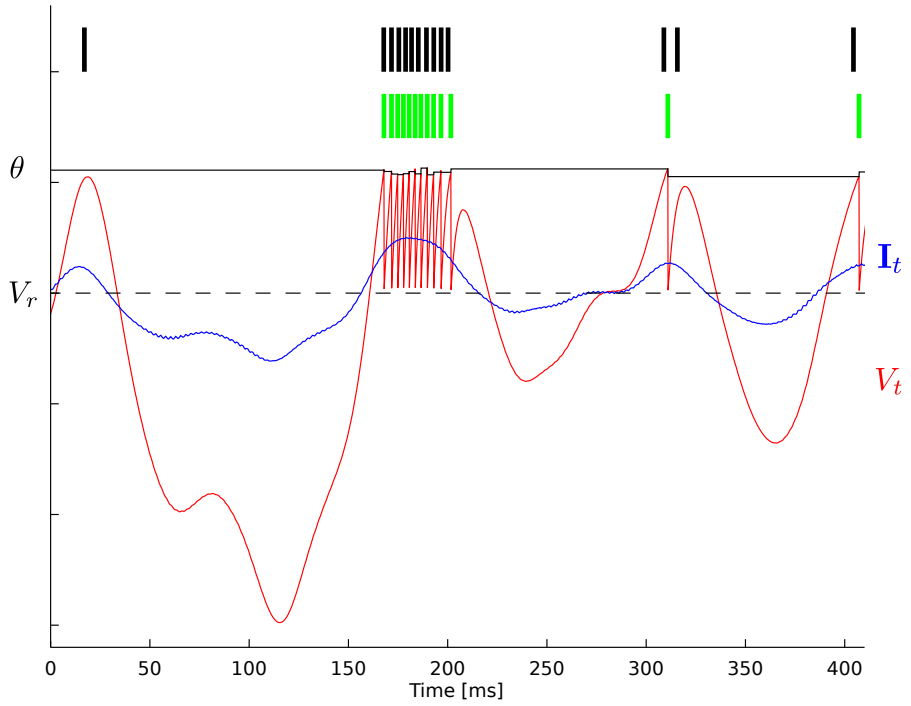

Figure 2.2: Encoding example for the case of threshold noise. The same input as for Figure 2.1 was used. The threshold, however was drawn from a Γ-distribution whenever a spike was fired. The mean and variance of the Γ-distribution were set such that the resulting spike train roughly resembles the one generated in the membrane noise setting. For comparison the spikes for the membrane noise are also plotted in black whereas the spikes generated in the threshold noise setting are plotted in green. The time constant and the effective input are the same as in Figure 2.1.

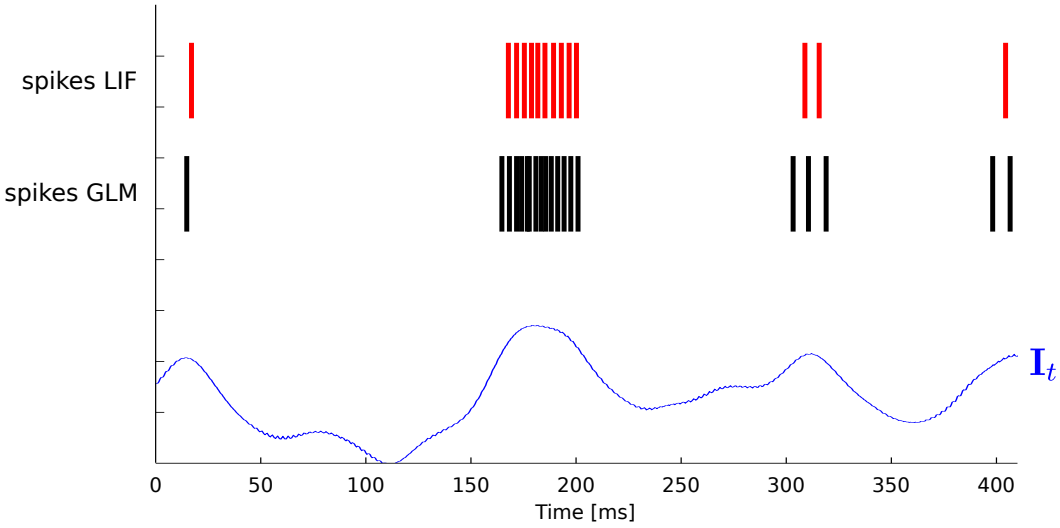

Figure 2.3: Encoding example using a GLM for spike generation. A GLM was stimulated with the same input as the one used in the Figure 2.1 and Figure 2.2. The receptive field for the stimulus and the linear filter for the spike-history were chosen such that the resulting spike-train resembles the one generated with the leaky integrator with membrane potential noise.

to include a linear filter acting on the recent spiking history of the neuron itself. In this way, neural properties such as refractory periods and bursting can be modeled. In addition, the spiking history of other neurons can be filtered as well. To adjust the noise level within the class of GLMs there are two possible ways. First, we can tune the link function. For example, we could set the link function f to:

$$f(x) := \begin{cases} 0, & \text{if } x \\ c, & \text{if } x \geq 0 \end{cases},$$

where c is a large constant value. This will result in a more deterministic spiking behavior, as the link function plays the role of a hard threshold. Secondly, we can allow for filters acting on the own spiking history. Finally, we could allow for combinations of the two possibilities to tune the degree of determinism.

As an illustration of the encoding, we simulated a GLM with the same stimulus that was used in the previous section and set the parameters of the GLM such that the produced spikes roughly matches the ones from the LIF, see also Chapter 3. In Figure 2.3 we plotted two sample spike trains generated by the GLM and the LIF respectively.

2.3 The maximum entropy model

The models in the previous sections can still be interpreted to roughly resemble some basic biophysical properties of neurons such as the membrane potential for the LIF. In this section, we further increase the level of abstraction. Instead of constraining the model by biologically motivated properties, we are here interested in finding a model which is only constrained by observed statistics. In terms of Occam's razor one is interested in finding the least structured model which is consistent with the data. We can formalize this by using (Shannon-) entropy which is a measure of structure of a probability distribution; a distribution with much entropy can thus be thought of as a simple distribution. Therefore, finding a distribution with maximal entropy which is still consistent with the data, then amounts to finding the simplest model.

The data consists of spike trains, which are sequences of discrete events in time. Within a small time bin there can either be a spike or no spike. Specifically, for n neurons we model the spikes in a time bin with a binary vector $\mathbf{b} \in \{1, -1\}^n$, where 1 corresponds to a 'spike'-event and -1 to a 'no-spike' event. Suppose, we have observed mean and (co-) variances of a set of neurons. Under all distributions $p(\mathbf{b})$ we can then find the one with maximal entropy which has the same observed moments. The distribution is given by:

$$\begin{aligned} p(\mathbf{b}|\mathbf{J}, \mathbf{h}) &= \frac{1}{Z} \exp\left(\mathbf{b}^\top \mathbf{J}\mathbf{b} + \mathbf{h}^\top \mathbf{b}\right) \\ Z &= \sum_{\mathbf{b} \in \{-1,1\}^n} \exp\left(\mathbf{b}^\top \mathbf{J}\mathbf{b} + \mathbf{h}^\top \mathbf{b}\right), \end{aligned} \tag{2.8}$$

where $\mathbf{J}, \mathbf{h}$ have to be chosen such that the moments of the distribution in equation (2.8) matches the observed ones. This model is also known as the Ising or Boltzmann model [Ising 1925]. Historically, the model was designed to model magnetic spins s_i of atoms. From a neuroscience perspective, atoms are usually interpreted as neurons, which can be either in a spiking (spin up) or silent (spin down) state in a particular time bin. Usually the stimulus dependence of the neurons is not modeled. However, in Chapter 5 we will show that the maximum entropy distribution with respect to the second order moments of pairs of binary and continuous variables can be calculated as well. The corresponding joint distribution over continuous stimulus variables $\mathbf{x}$ and binary neuronal variables $\mathbf{b}$ is given by:

$$p(\mathbf{x}, \mathbf{b}|\mathbf{\Lambda}, \boldsymbol{\lambda}) = \frac{1}{Z(\mathbf{\Lambda}, \boldsymbol{\lambda})} \exp\left(Q\left(\mathbf{x}, \mathbf{b}|\mathbf{\Lambda}, \boldsymbol{\lambda}\right)\right)$$

$$Q(\mathbf{x}, \mathbf{b}|\mathbf{\Lambda}, \boldsymbol{\lambda}) = \frac{1}{2} \begin{pmatrix} \mathbf{x} \\ \mathbf{b} \end{pmatrix}^{\top} \mathbf{\Lambda} \begin{pmatrix} \mathbf{x} \\ \mathbf{b} \end{pmatrix} + \boldsymbol{\lambda}^{\top} \begin{pmatrix} \mathbf{x} \\ \mathbf{b} \end{pmatrix}$$

$$Z(\mathbf{\Lambda}, \boldsymbol{\lambda}) = \sum_{\mathbf{b}} \int \exp\left(Q\left(\mathbf{x}, \mathbf{b}|\mathbf{\Lambda}, \boldsymbol{\lambda}\right)\right) \mathrm{d}\mathbf{x}$$

Fitting the parameters $\mathbf{\Lambda}, \boldsymbol{\lambda}$ to data is usually a hard problem, as the normalization constant Z cannot be evaluated analytically. Even the numerical computation of Z requires summing over exponentially (2^n) increasing number of terms. Analyzing this joint distribution is the subject of Chapter 5.

3

Encoding with generalized linear models

3.1 Introduction

In the encoding view, we model the mapping from stimuli to spikes. A classic description of this mapping can be obtained by the spike-triggered average [De Boer & Kuyper 1968, Marmarelis & Naka 1972]. Here, every time a spike is observed the preceding stimuli are collected and then the mean is computed. Theoretically, this can be motivated by a Volterra expansion, and therefore is a linear approximation to the encoding mapping [De Boer & Kuyper 1968, Dayan *et al.* 2001]. While this gives a good first order approximation of the encoding of a single neuron, the actual mapping is likely to be non-linear. Furthermore, the interaction between different neurons might also bias the estimated encoding of the spike-triggered average. This can be illustrated in the noiseless case: the response of two neurons with the same stimulus dependence could equally well be explained by one neuron depending on the stimulus, while the other has no stimulus dependence but is just copying the spikes of the first neuron.

Mathematically speaking, we would like to describe the probability of observing a particular response $\mathbf{r}$ given that we have presented a stimulus $\mathbf{s}$. More precisely,

because both the response and the stimulus vary over time, we are interested in predicting a whole spike-train $\mathbf{r}_t$ from a series of presented stimuli $\mathbf{s}_t$.

To obtain a characterization of this probability, two steps are needed. Firstly, we have to build a model which is sufficiently complex such that it can model the nonlinearities which are likely to be present. Secondly, we need tools for identifying the parameters of these models. In the previous chapter we introduced commonly used models. As a first step, all these models can include a linear filtering step. Hence, a primary problem is to find a suitable parametrization or representation of the stimulus space. For example, complex cells are invariant with respect to the exact position of the stimulus and only sensitive to the spatial frequency of the presented stimulus [Hubel & Wiesel 1962]. Hence, if we represent the stimulus in terms of the power spectrum, we can again predict the neural activity by a linear operation. However, finding such representations usually requires a non-linear operation, which is difficult to estimate. The most prominent methods for extracting the relevant feature spaces or parametrization are spike-triggered covariance techniques [Van Steveninck & Bialek 1988, Schwartz *et al.* 2002] or most informative dimensions [Sharpee *et al.* 2004]. This can also be extended to find features, to which not only a single neuron but a whole population is sensitive to [Macke *et al.* 2008].

The difficulty in choosing a model is to find the right trade-off between flexibility and tractability. Adding more parameters or features to the model makes it more flexible but also harder to fit, as it is more prone to overfitting. The Bayesian framework allows one to control for the model complexity even if the model parameters are underconstrained by the data, as imposing a prior distribution over the parameters allows regularizing the fitting procedure [Lewicki & Olshausen 1999, Ng 2004, Steinke *et al.* 2007, Mineault *et al.* 2009].

From a statistical point of view, building a predictive model for neural responses constitutes a regression problem. Linear least squares regression is the simplest and most commonly used regression technique. It provides a unique set of regression parameters, but one that is derived under the assumption that neural responses in a time bin are distributed according to a Gaussian. This assumption, however, is clearly not appropriate for the spiking nature of neural responses. Generalized linear models (GLMs) provide a flexible extension of ordinary least squares regression which allows on to describe the neural response as a point process [Chornoboy *et al.* 1988, Brillinger 1988] without losing the possibility of finding a unique best fit to the data [McCullagh & Nelder 1989, Paninski 2004].

The simplest example of the generalized linear spiking neuron model is the linear-nonlinear Poisson (LNP) cascade model [Chichilnisky 2001, Simoncelli *et al.* 2004]. In this model, one first convolves the stimulus with a linear

filter, subsequently transforms the resulting one-dimensional signal by a pointwise nonlinearity into a nonnegative time-varying firing rate, and finally generates spikes according to an inhomogeneous Poisson process. Importantly, the GLM model is not limited to noisy Poisson spike generation: Analogous to the stimulus signal, one can also convolve the recent history of the spike train with a feedback filter and transform the superposition of both stimulus and spike history filter outputs through the pointwise nonlinearity into an instantaneous firing rate in order to generate the spike output. In this way one can mimic dynamical properties such as bursts, refractory periods and rate adaptation. Finally, it is possible to add further input signals originating from the convolution of a filter kernel with spike trains generated by other neurons [Borisyuk *et al.* 1985, Chornoboy *et al.* 1988, Brillinger 1988]. This makes it possible to account for couplings between neurons, and to model data which exhibit so called noise correlations, i.e. correlations which can not be explained by shared stimulus selectivity. Although the GLM only gives a phenomenological description of the neurons' properties, it has been shown to perform well for the prediction of spike trains in the retina [Pillow *et al.* 2005, Pillow *et al.* 2008], in the hippocampus [Harris *et al.* 2003] and in the motor cortex [Truccolo *et al.* 2009].

In this chapter we seek to explore the potential uses and limitations of the framework for approximate Bayesian inference for GLMs based on the Expectation Propagation algorithm [Minka 2001]. With this framework, we can not only approximate the posterior mean but also the posterior covariance and hence compute confidence intervals for the inferred parameter values. Furthermore, the posterior mean is an alternative to the commonly used point estimators, maximum a posteriori (MAP) or maximum likelihood. Like the MAP also the posterior mean can be used with a Gaussian or a Laplacian prior leading to an L2 or an L1-norm regularization. To establish the approximate inference framework, we compare these point estimates on the basis of two different quality measures: prediction performance and filter reconstruction error. In addition, we investigate different binnings schemes and their impact on the different inference procedures. Along with the corresponding paper of this chapter we publish a MATLAB [1]

toolbox in order to support researchers in the field to do Bayesian inference over the parameters of the GLM spiking neuron model.

The chapter is organized as follows. In section 3.2, we review the definition of the generalized linear model and present the expansion into a high-dimensional feature space. We explain how a Laplace prior can improve the prediction performance in this setting and how different loss functions can be used to rate different quality aspects. In section 3.3, we present how the posterior distribution for observed

[1] the code is available at *http://www.kyb.tuebingen.mpg.de/bethge/code/glmtoolbox/*

data in the GLM setting can be approximated via the Expectation Propagation algorithm. Finally in section 3.4 we systematically compare the MAP estimator to the posterior mean assuming a Gaussian versus a Laplacian prior. In addition we apply the GLM framework to multi-electrode recordings from a population of retinal ganglion cells and discuss the potential differences of discretizing time directly or discretizing the features. Finally, we investigate the potential uses of the non-linear feature space. This includes finding confidence intervals for features resulting from a spike-triggered covariance analysis and approximating a leaky integrate-and fire neuron by using a customized feature space.

3.2 Generalized linear modeling for spiking neurons

3.2.1 Specifying the likelihood

The generalized linear model (GLM) of spiking neurons describes how a stimulus $\mathbf{s}(t)$ is encoded into a set of spike trains $\{t_j^i\}$ generated by neurons $i = 1, \ldots, N, j = 1, \ldots, N_i$ [Chornoboy *et al.* 1988, Brillinger 1988, Paninski 2004, Okatan *et al.* 2005, Truccolo *et al.* 2005] (See [Stevenson *et al.* 2008] for a recent review). More precisely, $\mathbf{s}(t)$ is a vector of dimensionality n, which describes the history of the stimulus signal up to time t according to a suitable parametrization. For example, in section 3.4 where we apply the GLM to retinal ganglion cell data, the vector $\mathbf{s}(t)$ contains the light intensities of the full-field flicker stimulus for the last n frames up to time t. The GLM assumes that an observed spike train $\{t_j\}$ is generated by a Poisson process with a time-varying rate $\lambda(t)$. In its simplest form the rate $\lambda(t)$ depends only on the stimulus vector $\mathbf{s}(t)$. This special case of the GLM is also known as the Linear-Nonlinear Poisson model [Simoncelli *et al.* 2004]. Specifically, the rate can be written as a Linear-Nonlinear cascade:

$$\lambda(t) = f(\mathbf{s}(t)^\top \mathbf{w}_s) \tag{3.1}$$

First, the stimulus is filtered with a linear filter $\mathbf{w}_s$ which is referred to as the *receptive field* of the neuron. Subsequently, the pointwise monotonic nonlinearity f transforms the real-valued output of the linear filtering into a nonnegative instantaneous firing rate. If the current stimulus has a strong overlap with the receptive field, that is if $\mathbf{s}(t)^\top \mathbf{w}_s$ is large, this will yield a large probability of firing. If it is strongly negative, the probability of firing will be zero or close to zero.

In the classical GLM framework [McCullagh & Nelder 1989], f is also called "link function". For the Poisson process noise model, the link function must be both convex and log-concave in order to preserve concavity of the log posterior

[Paninski 2004]. Thus it must grow at least linearly and at most exponentially. Typical choices of this nonlinearity are the exponential $f(x) = \exp(x)$ or a threshold linear function

$$f(x) = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{if } x \geq 0 \end{cases} .$$

As the spikes are assumed to be generated by a Poisson process, the log-likelihood of observing a spike train $\{t_j\}$ is given by

$$\begin{aligned} \log p(\{t_j\}|\mathbf{w}_s, \mathbf{s}(t)) &= \sum_j \log \lambda(t_j) - \int_0^T \lambda(\tau) \mathrm{d}\tau \\ &= \sum_j \log f(\mathbf{s}(t_j)^\top \mathbf{w}_s) - \int_0^T f(\mathbf{s}(\tau)^\top \mathbf{w}_s) \mathrm{d}\tau \quad . \end{aligned} \tag{3.2}$$

In this simple form, the GLM ignores some commonly observed properties of spike trains, such as refractory periods or bursting effects. In order to address this problem, we want to make the firing rate $\lambda(t)$ dependent not only on the stimulus but also on the history of spikes generated by the neuron. To this purpose, an additional linear filtering term can be added into equation (3.1). For example, by convolving the spikes generated in the past with a negative-valued kernel, we can account for the refractory period. The instantaneous firing rate of the GLM then results from a superposition of two terms, a stimulus and a spike feedback term

$$\lambda(t) = f(\mathbf{s}(t)^\top \mathbf{w}_s + \boldsymbol{\psi}_h(t)^\top \mathbf{w}_h) \quad . \tag{3.3}$$

The m-dimensional vector $\boldsymbol{\psi}_h(t)$ describes the spiking history of the neuron up to time t according to a suitable parametrization. A simple parametrization is a *spike histogram vector* whose components contain the number of spikes in a set of preceding time windows. That is, the $k-$th component $(\boldsymbol{\psi}_h(t))_k$ contains the number of spikes in the time window $(t - \Delta_{k+1}, t - \Delta_k]$ with $\Delta_0 < \Delta_1 < \cdots < \Delta_m$. The linear weights $\mathbf{w}_h$ can then be fit empirically to model the specific dynamic properties of the neuron such as its refractory period or bursting behavior. The encoding scheme is illustrated in Figure 5.1.

Analogous to the spike feedback just described, the encoding can readily be extended to the population case, if the vector $\boldsymbol{\psi}_h(t)$ for each neuron not only describes its own spiking history, but includes the spiking history of all other neurons as well. Taken together, the log-likelihood of observing the spike times $\{t_j^i\}$ for a population

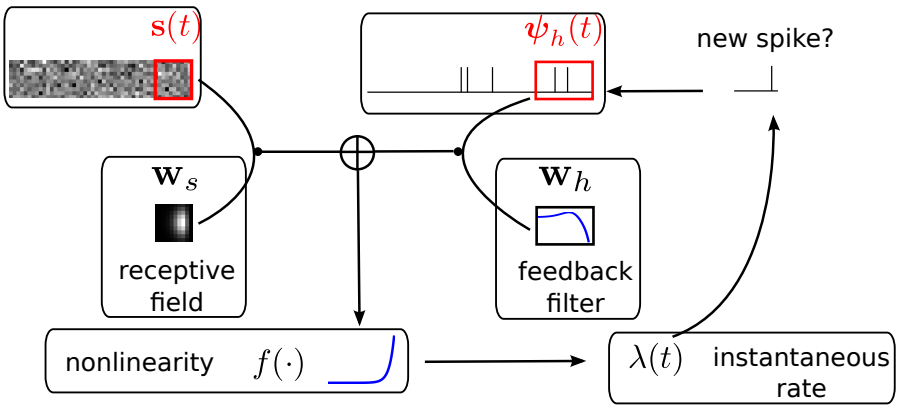

Figure 3.1: Illustration of the generative encoding model associated with a GLM: The stimulus $\mathbf{s}(t)$ as well as the spiking history $\boldsymbol{\psi}_h(t)$ are filtered with their corresponding receptive fields $\mathbf{w}_s$ and $\mathbf{w}_h$. A nonlinearity f is applied to the sum of the outputs to produce an instantaneous rate, which then is used to generate new spikes.

of $i = 1, \ldots, N$ neurons is given by

$$\begin{aligned} \log p(\{t_j^i\}|\mathbf{w}_s^i, \mathbf{w}_h^i) &= \sum_{i,j} \log \lambda^i(t_j^i) - \int_0^T \lambda^i(s)\mathrm{d}s \\ &= \sum_{i,j} \log f(\mathbf{s}(t_j^i)^\top \mathbf{w}_s^i + \boldsymbol{\psi}_h(t_j^i)^\top \mathbf{w}_h^i) \\ &\quad - \int_0^T f(\mathbf{s}(\tau)^\top \mathbf{w}_s^i + \boldsymbol{\psi}_h(\tau)^\top \mathbf{w}_h^i)\mathrm{d}\tau \quad . \end{aligned} \tag{3.4}$$

Although the likelihood factorizes over different neurons i, this does not imply that the neurons fire independently. In fact, every neuron can affect any other neuron i via the spiking history term $\boldsymbol{\psi}_h(t)$. Thus, by fitting the weighting term $\mathbf{w}_h^i$ to the data we can also infer effective couplings between the neurons.

In order to evaluate equation (3.4) we have to calculate the integral $\int_0^T f(\mathbf{s}(\tau)^\top \mathbf{w}_s^i + \boldsymbol{\psi}_h(\tau)^\top \mathbf{w}_h^i)\mathrm{d}\tau$ numerically. In terms of computation time, this easily becomes a dominating factor when the recording time T is large. Many artificial stimuli used for probing sensory neurons such as white noise can be described as piecewise constant functions. For example, the stimulus used for the retinal ganglion cells in section 3.4.3 had a refresh rate of 180 Hz. In this case, the stimulus $\mathbf{s}(t)$ only changes at particular points in time. Further, if we use the spike histogram vector mentioned above to describe the spiking history of the neurons, then also $\boldsymbol{\psi}_h(\tau)$ is a piecewise constant function. Thus, we can find time points $\tau_1, \ldots, \tau_z$ between which neither the stimulus nor the vector describing the spiking history

changes. We call the τ_i "discretization-points". Also in cases in which the features are not piece-wise constant such a discretization can be approximately obtained in a data-dependent manner, which we show in section 3.2.3. By decomposing the integral over $(0, T)$ into a sum of integrals over the intervals $[\tau_k, \tau_{k+1})$ within which the integrand stays constant, the log-likelihood can be simplified to

$$\begin{aligned}\log p(\{t_j^i\}|\mathbf{w}_s^i, \mathbf{w}_h^i) = &\sum_{i,j} \log f(\mathbf{s}(t_j^i)^\top \mathbf{w}_s^i + \boldsymbol{\psi}_h(t_j^i)^\top \mathbf{w}_h^i) \\ &- \sum_{k,i} (\tau_{k+1} - \tau_k) f(\mathbf{s}(\tau_k)^\top \mathbf{w}_s^i + \boldsymbol{\psi}_h(\tau_k)^\top \mathbf{w}_h^i)\end{aligned} \tag{3.5}$$

Note that $\boldsymbol{\psi}_h(\tau_k)$ and $\boldsymbol{\psi}_s(\tau_k)$ are constant, since the features do not change in the interval $[\tau_k, \tau_{k+1})$.

3.2.2 Extending the computational power of GLMs

To increase the flexibility of a GLM, several extensions are possible. For example, one can add hidden variables [Kulkarni & Paninski 2007, Nykamp 2008] or weaken the Poisson assumption to a more general renewal process [Pillow 2009]. By adding only a few extra parameters to the model these extensions can be very effective in increasing the computational power of the neural response model. The downside of this approach is that most of these extensions do not yield a concave log posterior anymore. Another option for increasing the flexibility of the GLM which preserves the desirable property of convexity is to add more and more linearly independent parameters for the description of the stimulus and spike history that are promising candidates for improving the prediction of spike generation. For example, in addition to the original stimulus components $\mathbf{s}(t)_i$ we can also include their quadratic interactions $\mathbf{s}(t)_i \mathbf{s}(t)_j$. In this way, we can obtain an estimate of the computations of nonlinear neurons such as complex cells. This is similar to the spike-triggered covariance method [Van Steveninck & Bialek 1988, Rieke *et al.* 1997, Rust *et al.* 2005, Pillow & Simoncelli 2006] but more general, as we can still include the effect of the spike history. In principle, one can add arbitrary features to the description of both the stimulus as well as the spiking history. As a consequence, it is possible to approximate any arbitrary point process under mild regularity assumptions (see [Daley & Vere-Jones 2008]).

Like in standard least squares regression the actual merit of the Bayesian fitting procedure described in this chpater is to have mechanisms for finding linear combinations of these features that provide a good description of the data. There-

fore, it often makes sense to use a set of basis functions whose span defines the space of candidate functions [Pillow *et al.* 2005]. We should choose a sufficiently rich ensemble of basis functions such that any plausible kind of stimulus or history dependence can be realized within this ensemble. We denote the feature space for the spiking history by $\boldsymbol{\psi}_h$ and the feature space for the stimulus by $\boldsymbol{\psi}_s$. The concatenation of both feature vectors is denoted by $\boldsymbol{\psi}_{s,h}$. Together we can write down the log-likelihood of observing a spike train $\{t_j^i\}_{j,i}$:

$$\log p(\{t_j^i\}|\mathbf{w}_s, \mathbf{w}_h) = \sum_{i,j} \log \lambda^i(t_j^i) - \sum_i \int_0^T \lambda^i(s)\mathrm{d}s \tag{3.6}$$

$$= \sum_{i,j} \log f(\boldsymbol{\psi}_h(t_j^i)^\top \mathbf{w}_h^i + \boldsymbol{\psi}_s(t_j^i)^\top \mathbf{w}_s^i)$$

$$- \sum_i \int_0^T f(\boldsymbol{\psi}_h(\tau)^\top \mathbf{w}_h^i + \boldsymbol{\psi}_s(\tau)^\top \mathbf{w}_s^i)\mathrm{d}\tau \tag{3.7}$$

3.2.3 Data-dependent discretization of the time-axis

If we choose the features $\boldsymbol{\psi}_h, \boldsymbol{\psi}_s$ such that they do not change between distinct discretization-points τ_k,i.e. $\boldsymbol{\psi}_{s,h}$ is constant in the interval $[\tau_k, \tau_{k+1})$ the likelihood can be simplified to:

$$\log p(\{t_j^i\}|\mathbf{w}_s, \mathbf{w}_h) = \sum_{i,j} \log f(\boldsymbol{\psi}_h(t_j^i)^\top \mathbf{w}_h^i + \boldsymbol{\psi}_s(t_j^i)^\top \mathbf{w}_s^i) \tag{3.8}$$

$$- \sum_{i,k} (\tau_{k+1} - \tau_k) f(\boldsymbol{\psi}_h(\tau_k)^\top \mathbf{w}_h^i + \boldsymbol{\psi}_s(\tau_k)^\top \mathbf{w}_s^i)$$

When approximating the features by describing the spike-history dependence with a piecewise constant function, this yields a finite number of discretization-points in time between which, the resulting conditional rate, given the spiking history, does not change. In order to illustrate this process, consider the following simple scenario illustrated in Figure 3.2. Suppose there is only one neuron, which receives a constant input. Accordingly, the feature describing the stimulus is constant $\boldsymbol{\psi}_s(t) \equiv 1$, which appear as the last entry in the combined feature vectors $\boldsymbol{\psi}_{h,s}(t)$ in the Figure. The spiking history H_t up to time t is represented by two dimensions, which are approximated by piece-wise constant functions, changing only at 2 and 10 ms. Note, that the time-axis, labeled with time-parameter s in Figure 3.2 is pointing into the past and centered at the current time-point t. As long as we did not observe a spike, the feature values of the two basis functions are zero, i.e. $\boldsymbol{\psi}_h(t)_1 = \boldsymbol{\psi}_h(t)_2 = 0$ for $t < t_1$. Once we have observed a spike, this enters in both features via the first constant value. Hence in this example $\boldsymbol{\psi}_h(t)_1 = 5, \boldsymbol{\psi}_h(t)_2 = 1$ for $\tau_1 = t_1 \leq t < \tau_2 = \tau_1 + 2\text{ms}$. When the observed spike leaves the 2 ms window

and enters the second time-window of the basis functions the feature values change to $\boldsymbol{\psi}_h(t)_1 = 1, \boldsymbol{\psi}_h(t)_2 = 2$ for $\tau_2 \leq t < \tau_3 = \tau_2 + 8\text{ms}$. In order to calculate the conditional rate, we have to evaluate $f(\boldsymbol{\psi}_h(t)^\top \mathbf{w}_h + \boldsymbol{\psi}_s(t)^\top \mathbf{w}_s)$. For the weights in Figure 3.2, this gives the qualitative time course of the conditional rate $\lambda(t|H_t, \mathbf{s}(t))$ as depicted in Figure 3.2.

3.2.4 Using Laplace priors for better regularization

The expansion of the stimulus and the spiking history in high-dimensional feature spaces comes at the cost of having a large number of parameters to deal with. As we only have access to a limited amount of data, regularization is necessary to avoid overfitting. In the Bayesian framework, this can be done by choosing a prior distribution $p(\mathbf{w}) = p((\mathbf{w}_s, \mathbf{w}_h))$ over the linear weights $\mathbf{w}_s$ and $\mathbf{w}_h$. As these parameters enter the log-likelihood linearly, the prior distribution can be interpreted as specifying how likely we think that a particular feature is active, or necessary for explaining a typical data set. The prior distribution becomes more important as we increase the number of parameters.

Two commonly used priors are the Gaussian

$$p(\mathbf{w}) = \frac{1}{2\sqrt{\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma}||\mathbf{w}||_2^2\right) = \frac{1}{2\sqrt{\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}\mathbf{w}^\top\mathbf{w}\right) \tag{3.9}$$

and the Laplace prior

$$p(\mathbf{w}) = \left(\frac{2}{\tau}\right)^n \exp\left(-\tau||\mathbf{w}||_1\right) = \prod_{k=1}^{n} \frac{2}{\tau} \exp\left(-\tau|w_k|\right). \tag{3.10}$$

Given a prior distribution, one can write down the posterior distribution

$$p(\mathbf{w}|D) \propto p(\mathbf{w})p(D|\mathbf{w})$$

which specifies how likely a set of weights $\mathbf{w}$ is, given the observed data D and the prior belief over the weights. The data D contains both, observed spike trains as well as stimuli.

To obtain a particular choice of parameter values a popular point estimate is maximum a posteriori (MAP) estimate, that is the point of maximal posterior density $\arg\max_{\mathbf{w}} p(\mathbf{w}|D)$. The MAP estimate is equivalent to the maximum likelihood estimate regularized with the log-prior. As mentioned above, the use of Laplace priors can yield advantageous regularization properties [Tibshirani 1996, Lewicki & Olshausen 1999, Ng 2004, Steinke *et al.* 2007, Mineault *et al.* 2009]. For a sparse prior, most of the features are likely to have zero weight, but if they have

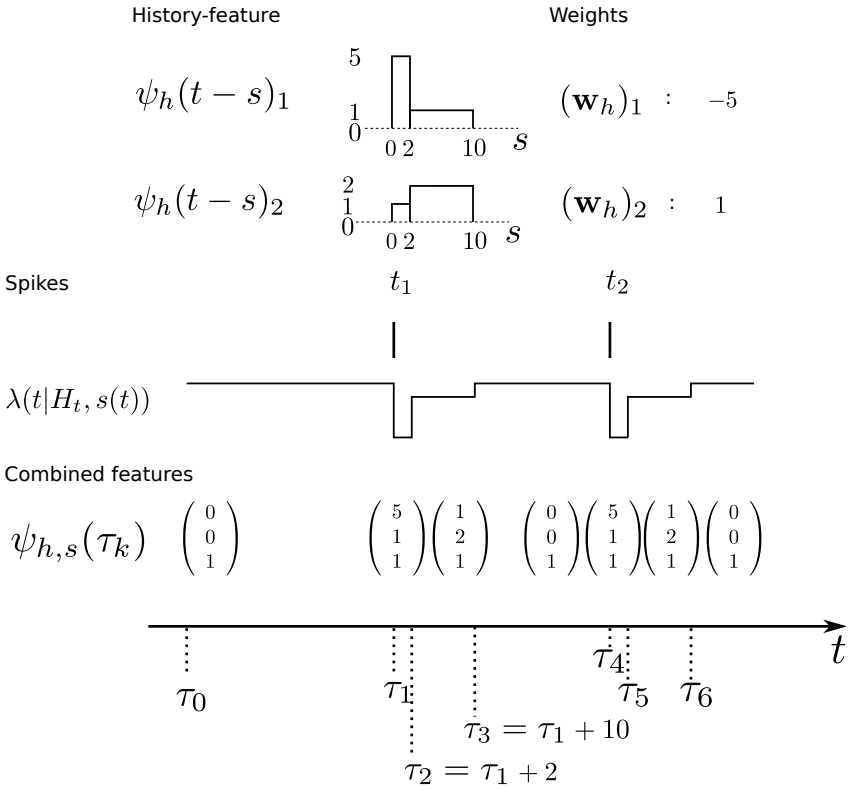

Figure 3.2: Illustration of the data-dependent time-discretization. Two spikes from one neuron have been observed at time-points t_1 and t_2. Since we assume a constant input ($\mathbf{s}(t) \equiv 1$) the last entry in the combined feature vector $\boldsymbol{\psi}_{h,s}(t)$ is always 1. The spiking history up to time t, denoted with H_t is described with two basis functions, $(\boldsymbol{\psi}_h(t))_1, (\boldsymbol{\psi}_h(t))_2$. Each of these could have its own discretization, but here both have the same, namely at 0 ms,2 ms and 10 ms. That is, the basis functions are approximated with a piecewise constant function with jumps at 0 ms,2 ms and 10 ms. Each spiking history feature has its own weight, as has the stimulus. Thus, the feature vector describing both, the stimulus as well as the spiking history $\boldsymbol{\psi}_{s,h}(t)$ is a 3-dimensional vector, changing its value at discretization points τ_k. In each interval $[\tau_k, \tau_{k+1})$ the rate $\lambda(\tau_k|H_t, \mathbf{s}(\tau_k))$ can be calculated. In this specific case, it only assumes 3 different values, $\exp(\boldsymbol{\psi}_{s,h}(t)^\top \mathbf{w}_{h,s}) = \exp(-27), \exp(-2), \exp(1)$, assuming that the weight for the stimulus is $\mathbf{w}_s = 1$.

a non-zero weight, the amplitude is less constrained. In order to favor sparse solutions, the direct approach would be to penalize the number of non-zero parameter entries. The number of non-zero entries is sometimes referred to as the "L0-norm" of the parameter vector (despite the fact that it is not a proper norm). Unfortunately, finding the L0-norm regularized weights is a hard problem. Using the L1-norm however, is a useful relaxation which in some cases even gives an equivalent solution [Donoho & Stodden 2006]. The log of the Laplace prior-probability (see equation 3.10) of a given parameter vector is proportional to the L1-norm of this vector. Therefore, using a Laplace prior is equivalent to penalizing the L1-norm of the parameters. Finally using a Gaussian prior is equivalent to penalizing the L2-norm of the parameter vector (see equation 3.9).

From a practical point of view, log-concavity is another desirable property of the prior distribution as it here ensures that the posterior $p(\mathbf{w}|D) \propto p(\mathbf{w})p(D|\mathbf{w})$ is also log-concave and therefore finding the maximum of the posterior (i.e. computing the MAP estimator) is a convex optimization problem [Paninski *et al.* 2004]. For the GLM, log-concavity and convexity of the link-function f is also required to guarantee log-concavity of the posterior. Both priors, the Gaussian as well as the Laplacian are log-concave. Although the posterior is log-concave when a Laplace prior is used, calculating the MAP is still a non-trivial problem. As the Laplace prior is non-differentiable at zero, the gradient at any point containing a zero in at least one component cannot be calculated. Thus standard techniques like conjugate gradient or iterative reweighted least squares fail. For the case of a Gaussian likelihood and Laplace prior the LASSO algorithm [Tibshirani 1996] can be used. For the case of a likelihood originating from a GLM, the posterior is differentiable in each orthant, and hence subgradients can be calculated. In our implementation, we use the algorithm of Andrew et al. [Andrew & Gao 2007].

3.2.5 Quantifying the performance

After we have obtained an estimate of the parameters of a GLM, we would like to evaluate the quality of the estimate.

Prediction Performance To measure the performance of an estimate, we calculated the difference between the estimated model and the ground truth model with respect to the log likelihoods on a test set. The test set was generated with the same weights for each trial. In this way we can assess how likely a previously unseen spike train sampled from the ground truth model is under the estimated model. The difference between the average log likelihoods yields an estimate of Kullback-Leibler distance of the estimated model from ground truth.

$$l(\mathbf{w}, \hat{\mathbf{w}}) = \frac{1}{N}\sum_{i=1}^{N} \log p(D_i|\mathbf{w}) - \log p(D_i|\hat{\mathbf{w}}) \approx \int \log\left(\frac{p(D|\mathbf{w})}{p(D|\hat{\mathbf{w}})}\right) p(D|\mathbf{w})\mathrm{d}D \\ = D_{\mathrm{KL}}[p(\cdot|\mathbf{w})||p(\cdot|\hat{\mathbf{w}})] \tag{3.11}$$

Here D_i is a spike train in the i-th of N trials generated with the true weights $\mathbf{w}$ whereas the estimated weights are $\hat{\mathbf{w}}$. The more likely the spike trains are, the better is the weight estimate, which specifies the estimated model. Therefore, the difference in log likelihood of the different models measures how well the estimated model predicts the spike times generated by the ground truth model.

In the simplest (LNP) case, the parameters $\mathbf{w}, \hat{\mathbf{w}}$ correspond to rates $\lambda, \hat{\lambda}$ of a Poisson distribution, i.e. the probability of observing n spikes within a small time bin δt is proportional to $\lambda, \hat{\lambda}$. In this case we can calculate the loss function explicitly:

$$\begin{aligned} l(\mathbf{w}, \hat{\mathbf{w}}) &= D_{\mathrm{KL}}\left[p(n|\lambda)||p(n|\hat{\lambda})\right] \\ &= \sum_n \left(n\log(\lambda) - \log(n!) - \lambda - n\log(\hat{\lambda}) + \log(n!) + \hat{\lambda}\right)\frac{\lambda^n}{n!}\exp(-\lambda) \\ &= \lambda\log(\lambda) - \lambda - \lambda\log(\hat{\lambda}) + \hat{\lambda} \end{aligned} \tag{3.12}$$

To illustrate the asymmetry of the loss function the average log-loss is shown in Figure 3.3. The average log-loss is closely related to the Kullback-Leibler distance:

$$D_{\mathrm{KL}}\left[p(n|\lambda)||p(n|\hat{\lambda})\right] = \langle -\log(p(n|\hat{\lambda})\rangle_\lambda - H[p(n|\lambda)] \tag{3.13}$$

where H is the entropy of the Poisson distribution with rate λ.

Mean Squared Error Reconstruction A different way of quantifying the performance of an estimation algorithm for synthetic data is to check how closely the estimated parameters ($\hat{\mathbf{w}}$) match those that were put into the model as ground truth ($\mathbf{w}$). In particular for judging the quality of the reconstructed filter shapes a possible choice is to look at the mean square error between the true and estimated parameters:

$$l(\mathbf{w}, \hat{\mathbf{w}}) = \sum_j |\mathbf{w}_j - \hat{\mathbf{w}}_j|^2 \tag{3.14}$$

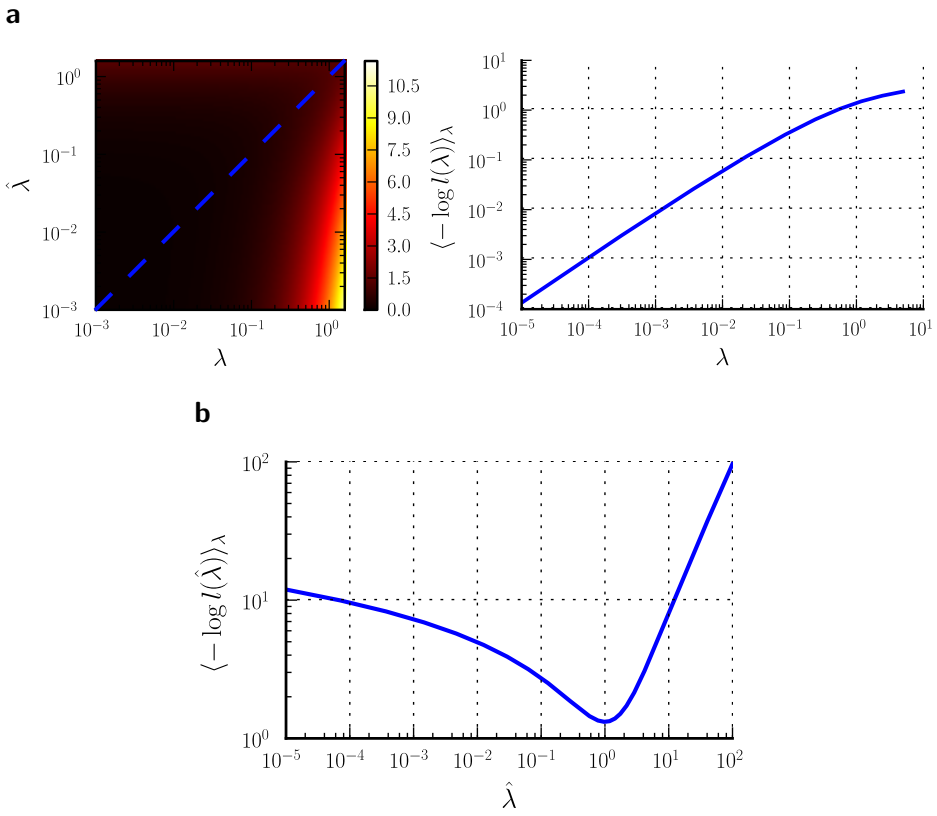

Figure 3.3: Comparison of estimated versus true underlying rate of a Poisson distribution. Plotted is the average log-loss as a function of the estimated rate $\hat{\lambda}$, although the true underlying rate is λ. **Upper left:** The average log-loss is plotted color-coded as a function of both rates. **Upper right:** The entropy of a Poisson distribution is plotted as a function of the underlying rate λ. It can also be seen as a slice of the left plot across the diagonal. **Bottom:** A vertical cut of panel **a** at $\lambda = 1$ is plotted as a function of the estimated rate $\hat{\lambda}$. The minimal loss can be achieved, if the estimated rate matches the true rate. Furthermore, if the estimated rate differs from the true rate by a fixed amount, the loss is smaller if the rate is overestimated.

3.3 Approximating the posterior distribution using EP

It has been shown that the MAP yields a good prediction performance [Pillow *et al.* 2008] but there are a couple of reasons why one would like to know more about the posterior than just its maximum. For example the posterior mean is known to be the optimal point estimate with respect to the mean squared error ((3.14)). Furthermore, in many cases we are not only interested in a point estimate of the parameters, but we also want to know the dispersion of the posterior. In other words, we want to have confidence intervals indicating how strongly the

parameters of a model are constrained by the observed data.

The resulting uncertainty estimate in turn can be used for optimal design [Lewi *et al.* 2008, Seeger 2008], that is we can decide which stimulus to present next, in order to maximally reduce our uncertainty about the parameters. Furthermore, a distribution of the full posterior distribution gives rise to the marginal likelihood, which is the likelihood of the data under the model, without assuming specific linear filters. The marginal likelihood can be used to optimize the parameters of the prior without performing a crossvalidation [Chib 1995, Seeger 2008]. Mathematically, the uncertainty is encoded in the dispersion of the posterior distribution over parameters $\mathbf{w}$ given observed data D:

$$p(\mathbf{w}|D) = \frac{1}{Z} p(D|\mathbf{w})p(\mathbf{w}) \tag{3.15}$$
$$\text{where} \quad Z = \int p(D|\mathbf{w})p(\mathbf{w})\mathrm{d}\mathbf{w}.$$

Taken together there are good reasons why it is useful to investigate the information conveyed by the posterior other than just the location of its maximum. The posterior really is the summary of everything we can learn from the data about the given model.

Unfortunately, exact Bayesian inference is intractable in our case. Therefore, we are interested in finding a good approximation to the full posterior. If we can determine the posterior mean and covariance, this naturally leads to a Gaussian approximation of the posterior. Furthermore, we note that the true posterior in our case is unimodal, as both likelihood and prior are log-concave [Paninski 2004]. We employ the Expectation Propagation (EP) algorithm in order to compute a Gaussian approximation to the full posterior [Minka 2001, Opper & Winther 2000, Opper & Winther 2005, Seeger 2005] (see [Nickisch & Rasmussen 2008] for alternative approximations schemes). The key observation is that the likelihood as well as the Laplace prior factorizes over simple terms, each of which is intrinsically one-dimensional. We have three types of factors

$$f_1(u_i) = \exp(\log(f(u_i)) - \Delta\tau_i f(u_i)) = f(u_i)\exp(-\Delta\tau_i f(u_i)) \tag{3.16}$$
$$f_2(u_i) = \exp(-\tau_i f(u_i)) \tag{3.17}$$
$$f_3(u_i) = \exp(-\tau|u_i|) \tag{3.18}$$

where, $u_i := \boldsymbol{\psi}_{s,h}(\tau_i)^\top \mathbf{w}_{s,h}$ defines the one-dimensional direction for each of these factors. $\boldsymbol{\psi}_{s,h}$ and $\mathbf{w}_{s,h}$ denote the concatenation of the feature vectors describing the spiking history and the stimulus history respectively. Equation 3.16 corresponds to a factor or individual term in the sum of the log likelihood (5.8) if there was a spike

at τ_{i+1} and no spike in the interval (τ_i, τ_{i+1}) of length $\Delta\tau_i := (\tau_{i+1} - \tau_i)$. Equation 3.17 corresponds to a factor if there was no spike at time τ_{i+1}. Finally equation 3.18 represents the Laplace terms for the prior in the product for the posterior distribution. The Expectation Propagation algorithm approximates each of those factors with a Gaussian factor:

$$f_i(u_i) \approx \exp\left(-\frac{1}{2}\pi_i u_i^2 + b_i u_i\right) \tag{3.19}$$

Thus, if we multiply all of these approximating factors, we obtain a Gaussian distribution, which is straightforward to normalize:

$$p(\mathbf{w}|D) \approx Q(\mathbf{w}) := \frac{1}{Z}\prod_i \exp\left(-\frac{1}{2}\pi_i u_i^2 + b_i u_i\right) \tag{3.20}$$

$$= \frac{1}{Z}\exp\left(-\frac{1}{2}\mathbf{w}^\top \sum_i \pi_i \boldsymbol{\psi}_{s,h}(\tau_i)\boldsymbol{\psi}_{s,h}(\tau_i)^\top \mathbf{w} + \sum_i b_i \boldsymbol{\psi}_{s,h}(\tau_i)^\top \mathbf{w}\right) \tag{3.21}$$

$$= \frac{1}{(2\pi)^{n/2}\det \mathbf{C}^{1/2}}\exp\left(-\frac{1}{2}\left(\mathbf{w}-\boldsymbol{\mu}\right)^\top \mathbf{C}^{-1}\left(\mathbf{w}-\boldsymbol{\mu}\right)\right) \tag{3.22}$$

where (3.23)

$$\mathbf{C} = \left(\sum_i \pi_i \boldsymbol{\psi}_{s,h}(\tau_i)\boldsymbol{\psi}_{s,h}(\tau_i)^\top\right)^{-1} \tag{3.24}$$

$$\boldsymbol{\mu} = \mathbf{C}\left(\sum_i b_i \boldsymbol{\psi}_{s,h}(\tau_i)\right) \tag{3.25}$$

The task now is to update the parameters π_i, b_i for the approximating factors such that the moments of the resulting approximation are as close to the true moments as possible. The crucial consistency equation which the EP algorithm tries to attain is given by[Opper & Winther 2005]:

$$D_{\mathrm{KL}}\left[\, f_i(u_i)\frac{Q(u_i)}{\exp(-\frac{1}{2}\pi_i u_i^2 + b_i u_i)}\,\middle\|\, Q(u_i)\right] \stackrel{!}{=} 0, \tag{3.26}$$

where D_{KL} denotes the Kullback-Leibler divergence or relative entropy. $Q(u_i)$ is the marginal Gaussian distribution in the direction of $\boldsymbol{\psi}_{s,h}(\tau_i)$. It is the Gaussian distribution one obtains, when taking the complete approximation $Q(\mathbf{w})$ and projects it on $\boldsymbol{\psi}_{s,h}(\tau_i)$. In other words, we require the approximation to be consistent in the sense that, if we replace the approximating factor $\exp(-\frac{1}{2}\pi_i u_i^2 + b_i u_i)$ with the true factor $f_i(u_i)$, the marginal moments in the direction of $\boldsymbol{\psi}_{s,h}(\tau_i)$ should not change. To achieve this consistency, EP cycles through the factors and updates the parameters of each approximating factor such that equation 3.26

holds. For equation 3.26 to hold, only moments of a one-dimensional distributions have to be calculated. This can efficiently be done using numerical integration [Piessens *et al.* 1983]. We omit the details of this updating scheme here and refer to the Appendix. The interested reader is referred to our MATLAB code and to further literature [Heskes *et al.* 2002, Qi *et al.* 2004]. The computational cost of EP is quadratic in the number of parameters (as the posterior covariance has to be estimated) and linear in the number of factors (in the GLM setting this is the same as the number of discretization-points) per cycle through the factors. In our simulations 30 iterations through all factors were sufficient for convergence.

Another frequently used way of approximating the posterior distribution with a Gaussian, is the so called Laplace approximation or Laplace's method [MacKay 2003, Rasmussen & Williams 2006, Lewi *et al.* 2008]. A second-order Taylor expansion is calculated around the MAP. As the posterior is unimodal, the MAP can be found efficiently. Calculating the Hessian at a particular point can also be obtained analytically, given the posterior is differentiable at that point. The Laplace prior we use, however, is non-differentiable at zero. Therefore, the posterior is not differentiable at any point which contains at least one zero in one component. As we expect the MAP to assign many components zero weight, we cannot calculate the Hessian at that point. Furthermore, in a different setting it has been shown that the quality of the Laplace approximation is inferior to the one achieved by the EP approximation [Kuss & Rasmussen 2005, Koyama & Paninski 2009]. The Laplace approximation is only sensitive to the local curvature at the point of maximal posterior density. As the EP-approximation is based on moment matching it is influenced by the shape of the full posterior distribution.

3.4 Potential uses and limitations

In the following, we systematically compare the different point estimates, posterior mean and MAP. We vary the assumed prior distribution as well as the loss function in terms of which the performance is measured. In particular, we also investigate cases in which the assumed prior distribution differs from the 'true' distribution used to generate the parameters. Finally, we also look at the possible effects of data discretization.

3.4.1 Maximum a posteriori vs. posterior mean

Tibshirani [Tibshirani 1996] showed that for Gaussian likelihood and Laplace priors, the MAP gives sparse solutions and performs best, given the true underlying weights

are sparse. If the data is assumed to be distributed according to a logistic likelihood, a similar result has been found by Ng [Ng 2004]. Here, for the case of data generated by a GLM, we would like to see whether the same holds true, and also compare the MAP to the posterior mean.

To illustrate the effect of a Laplace prior when increasing the number of features in the GLM of spiking neurons, we considered the following examples. We made a series of simulations with GLM neurons for which the space of possible features was successively increased from 10 to 230 dimensions. The stimulus was Gaussian white noise discretized into 10 ms bins. The stimulus history $\mathbf{s}(t)$ was set to contain the stimulus values of the last twenty bins describing the stimulus history for a period of 200 ms. From the 20 dimensional stimulus history $\mathbf{s}(t)$ we constructed the full 230 dimensional quadratic feature space

$$\begin{aligned}\boldsymbol{\psi}_s(t) :=&(s(t), \ldots, s(t-20\Delta),\\ &s(t)^2, s(t)s(t-\Delta), \ldots, s(t)s(t-20\Delta),\\ &s(t-\Delta)^2, s(t-\Delta)s(t-2\Delta), \ldots,\\ &\ldots, s(t-20\Delta)^2)\end{aligned}$$

with $\Delta = 10$ms, similar as in [Rust *et al.* 2005]. From this basis of the 230 dimensional feature space a subset of increasing size was selected. That is, the dimensionality of the weight vector increased from 10 to 230, too. For all simulations, a GLM neuron was simulated until the likelihood consisted of 400 factors, i.e. 400 τ_k in the sum in equation 3.8 (alternatively one could also fix the time-duration of a trial or the number of spikes per trial).

We compared three different choices of priors, and use models which either had matching priors, or different ones:

1. **Gaussian weights** Each weight was sampled independently from a Gaussian distribution. The variance was set to $\frac{20}{\dim(\psi_s)}$.
2. **Laplacian weights** Each weight was sampled independently from a Laplace distribution. The variance was set to $\frac{20}{\dim(\psi_s)}$.
3. **Sparse weights** A subset of only 10 dimensions was assigned with non-zero weights. For the assignment of the 10 weights, we draw 10 samples from a Laplace distribution with variance 2 and zero mean.

In Figure 3.4 the Kullback-Leibler distance is plotted as a function of the dimensionality of the feature space for each of the generating distributions. In Figure 3.4

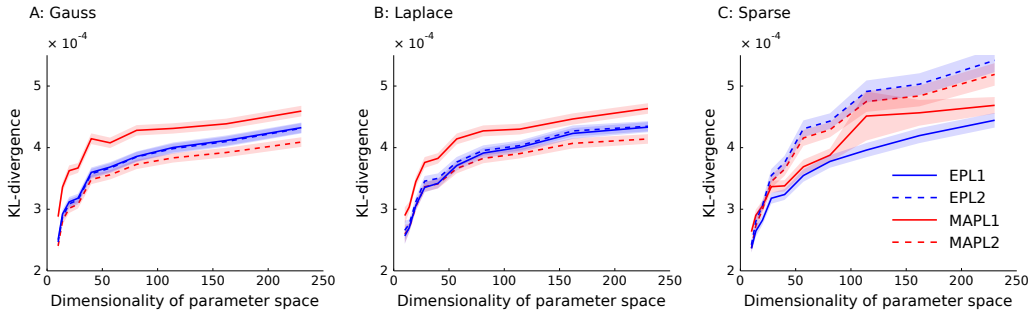

Figure 3.4: Prediction performance in high dimensional feature spaces of increasing size. The mean across 5000 trials of the differences in the log likelihoods is plotted as a function of increasing stimulus dimension. The different point estimates are MAP with Laplace regularization (MAPL1, solid red), MAP with a Gaussian prior (MAPL2, dashed red) and the posterior mean approximated with EP for the Laplace (solid blue) as well as for the Gaussian prior (dashed blue). Confidence intervals indicate standard error of the mean difference. Panel A) shows the performance when a Gaussian distribution is used for sampling the weights and B) for a Laplace distribution. C) shows the prediction performance if the weights are actually sparse, that is the true dimensionality is constantly 10. The overall variance for the generation of weights in panel A) and B) were kept fix to the same value as in C).

A) the weights of the ground truth model are sampled from a Gaussian distribution. Analogously, panel B) shows the results for the Laplace distribution and C) for the strongly sparse weights. We plot the average KL-divergence over 5000 trials $\pm$ one standard deviation. As can be seen, the EP estimate for the Laplace (L1) prior performs best, if the true underlying weights are sparse. If the weights are sampled from a Laplace or a Gaussian distributions, the parameter vector of the true model is non-sparse and the L2 regularized MAP performs best. Interestingly, even for the case in which the weights are sampled from a Laplace distribution, the MAP performs best when using an L2-penalty term. Since we know the prior variance that was used to generate the weights, we did not perform a crossvalidation to set the regularization parameter, neither for the MAP estimates, nor for the posterior mean estimates (EPL1,EPL2). (Note that, in cases where the true distribution of weights is different to the prior used, it is possible that the prediction performance could be increased by picking a variance which is different to the 'true' one.)

In cases, in which the parameters are really drawn from the prior distribution, the posterior mean estimate can be shown to be the optimal parameter estimate, as it will minimize the mean squared error. Thus, in the two cases, in which we sampled the weights according to a Gaussian and a Laplacian distribution respectively, we expect the EP-approximation to be superior to the MAP estimate in terms of the mean squared error. In the situation where the weights are actually sparse the

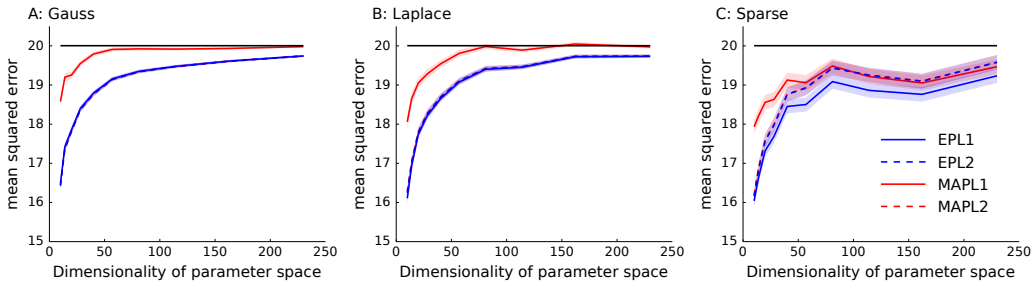

Figure 3.5: Mean squared error as a function of increasing dimensionality of the parameter space. The same data as in Figure 3.4 is plotted, but instead the performance is measured in mean squared error between the estimated weights and the true underlying weights as opposed to the differences in log likelihoods shown in Figure 3.4. A) shows the performance if the underlying weights are sparse, in panel B) a Laplace distribution is used to sample the weights and in C) a Gaussian distribution is used. In each panel the mean across 5000 trials is plotted ± standard error of mean. In solid black the prior variance is plotted, which is the expected mean squared error of the constant estimator.

performance is less clear, as the EP estimates assume a prior which is different to the one used to generate the weights. Therefore, it is not guaranteed in this case, that the posterior mean will be the optimal parameter estimate with respect to the mean squared error.

In general, we expect the MAP estimate to give a sparser solution than the posterior mean. If we have not seen much data, we expect the prior to dominate the posterior. In this case the maximum of the posterior will be at zero, resulting in a zero weight for the MAP. However, as the likelihood factors are not symmetric, the posterior is also not symmetric in general. Thus, even for weights for which the MAP is at zero, the probability mass is not symmetrically distributed around that maximum. Hence, the posterior mean in this case will be non-zero and the solution less sparse. In Figure 3.5 we plotted the mean squared reconstruction error for the different estimators. As can be seen the EP approximation to the posterior mean performs better than the MAP. This is also true for the sparse setting, however the effect gets less prominent if the dimensionality of the parameter space is increased.

The quality of the different point estimates, quantified by the mean squared error and by the prediction performance are summarized in Table 3.1. To obtain a single number for the overall performance, we summed the errors for each individual dimension of parameter space (integral over each curve in Figure 3.4 and Figure 3.5). The posterior mean gives a good estimate in all settings when a Laplacian prior is used. For the prediction performance the MAP with the L2 prior can lead to better results if the true prior is Gaussian or Laplacian.

		Integrated KL-divergence				Integrated MSE			
		MAP with		EP-mean with		MAP with		EP-mean with	
		Laplace	Gauss	Laplace	Gauss	Laplace	Gauss	Laplace	Gauss
Ground truth	Gauss	$3.93\ 10^{-3}$	**$3.39\ 10^{-3}$**	$3.532\ 10^{-3}$	$3.5\ 10^{-3}$	195.996	186.095	186.248	**185.992**
	Laplace	$3.87\ 10^{-3}$	**$3.46\ 10^{-3}$**	$3.52\ 10^{-3}$	$3.58\ 10^{-3}$	194.246	185.52	**184.99**	185.391
	Sparse	$3.66\ 10^{-3}$	$3.83\ 10^{-3}$	**$3.41\ 10^{-3}$**	$3.96\ 10^{-3}$	188.698	183.685	**180.536**	183.542

Table 3.1: Comparison of different quality measures and point estimates. In the left table integrated KL-divergence is shown for the MAP and the posterior mean point estimates when either a Laplace or a Gaussian prior is assumed. Each row corresponds to a ground truth prior which was used to sample the weights. Each number corresponds to an integral of a curve in Figure 3.4. The right table reports the same when the mean squared error is used as a loss function. Thus each number is the integral over one curve in Figure 3.5 and therefore reports the overall performance of the different estimators. For each ground truth model and loss function the best overall estimator is colored in red.

3.4.2 Binning and identifyability

In section 3.2 we specified the log-likelihood in terms of time-discretized features. This results in a binning with not necessarily equidistant discretization-points τ_j. Another popular way to simplify the log-likelihood is to bin the time axis directly. In this section we would like to illustrate the possible effects of the two discretizations by means of a simple example. For some areas, for example in the auditory cortex, the precise timing of spikes is important [Carr & Konishi 1990, Wightman & Kistler 1992]. By binning spikes into a discrete set of bins, one might lose this precise timing. If one discretizes the time axis directly and wants to keep the precise timing, one needs to specify very small time bins. This leads to a large number of discretization-points and hence very many factors for the likelihood. Alternatively, if one discretizes the features, the discretization is adapted to the spike times and thus could lead to possibly fewer discretization-points while still achieving a high temporal resolution. However, if a lot of spike times have been observed, discretization of the basis functions for the features could lead to a time discretization which is too fine for optimization purposes. A compromise would be to adaptively add discretization-points when needed, but constrain the minimal inter discretization-point interval. In general, the discretization of the features allows one to specify the resolution and (given that resolution) produces then the minimal number of discretization-points.

To illustrate possible differences between a discretization of features versus a discretization of the time axis, we considered the following example: Two GLM

neurons were simulated. One of them had a stimulus filter, while the other one was only dependent on the spikes from the first neuron. The filters for the stimulus as well as the spiking history filters are illustrated in Figure 3.6 (black lines). Because the second neuron was positively coupled to the first one with a small latency, we expect it to produce spikes which have a small temporal offset with respect to the spikes of the first neuron. Intuitively, the observed spikes trains could be explained by two different settings:

1. The weights are exactly as the ones used for simulating the spike trains.

2. The second neuron is not coupled to the first neuron at all, but has the same stimulus filter as the first one, however with a small latency. Therefore it responds to the same stimulus but at later times.

If spikes were generated deterministically, these two setting cannot be distinguished. In the noisy case, however, given a sufficient amount of data, one should be able to disentangle the two scenarios, as finding the maximum likelihood point is a convex problem. However, for finite amount of training data and in the presence of binning noise, the situation is less clear. Therefore, we sampled 3 seconds of spike trains and estimated the parameters from the data, once when the features are discretized and once when the time axis is discretized. The time bins were chosen such that at most one spike fell into a bin.

The estimate for the approximated posterior mean are plotted in Figure 3.6. If the features are discretized the filter could be recovered. If we discretize the time directly, we see indeed a slight shift towards the second scenario. That is, the stimulus filter for the second neuron in that case is slightly elevated, whereas the strength of the coupling filter is diminished.

3.4.3 Population of retinal ganglion cells

To compare the different methods for the analysis of real data, we applied the algorithms to multi-electrode recordings of 7 salamander retinal ganglion cells. Our goal was to describe the stimulus selectivity of the population by fitting a GLM with history terms and cross-neuron terms to the recorded data. We used multi-electrode recordings of salamander retinal ganglion cells generously provided by Michael J Berry II. The dataset has been published in [Fairhall *et al.* 2006], where all recording details are described. We selected a recording of 7 neurons, which had an average firing rate of 1.1 spikes per second and a minimal interspike-interval of 2.8 ms. The stimulus used in the experiments consisted of 20 minutes white noise full-field flicker with a refresh rate of 180 Hz. To illustrate the ability of the

A) B)

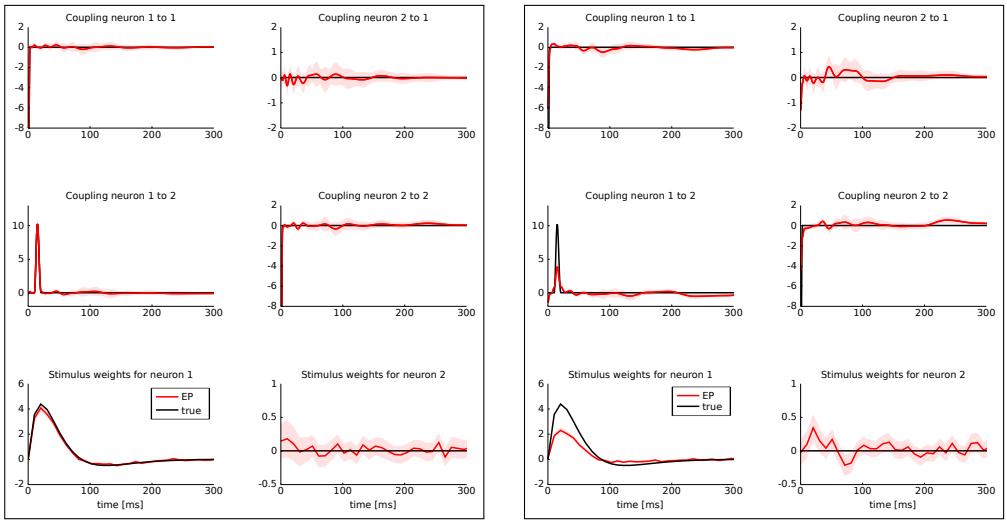

Figure 3.6: Identifyability in the presence of binning noise. A): Estimated filters, when the features are discretized (approximated with a piecewise constant function, see Figure 3.2). B): Estimated filters when the spike times are binned. The binning was performed such that at most one spike fell into one bin. All spikes were aligned to the right hand side of their corresponding bins. When the time-axis is binned directly and hence the precise timing of a spike is lost, the estimated filter for the spiking history are slightly weaker than the true ones (black), whereas the stimulus filters are slightly positive at a small latency. For the sake of readability we only plotted the approximated posterior mean ($\pm 2\sigma$).

model to also infer population models from small data sets, we fitted the population recording to the first 2 minutes of the recording.

For the features describing the spiking history, we used the density function of the Γ-distribution with different parameters as basis functions:

$$f_i(t) = t^{\alpha_i - 1} \exp(-\beta_i t) \frac{\beta_i^{\alpha_i}}{\Gamma(\alpha_i)}, \tag{3.27}$$

where the means $\frac{\alpha_i}{\beta_i}$ as well as the variances $\frac{\alpha_i}{\beta_i^2}$ were logarithmically spaced between 1 and 700 ms and 1 and 1000 respectively (A similar basis consisting of raised cosines was also used in [Pillow *et al.* 2005, Pillow *et al.* 2008]). Due to the logarithmic spacing, we have a finer resolution for small time-lags and coarser resolution for long time-lags. For example, we expect the first basis function, which has a sharp peak at zero to be mainly active or associated with the refractory period. As we discretize

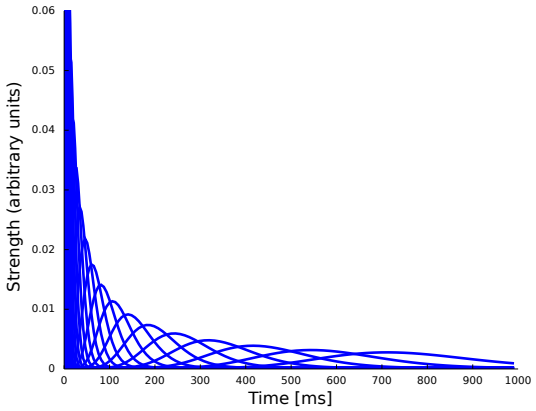

Figure 3.7: Set of 23 basis functions to span the spiking history as well as the stimulus dependence. Each function is a density-function of a Γ-distribution with different means and variances, see equation (3.27). The time-axis for the features describing the spiking history was logarithmically discretized up to 1000 ms.

the basis functions rather than directly the time-axis, each spike generates as many discretization-points τ_j as there are discretization points for the basis functions (see Section 3.2). For the stimulus we used the same basis function set. As for the spike-history dependence these functions were approximated with a piecewise constant function. The discretization for the basis-function time-axis in this case was the same as for the original stimulus and therefore slightly coarser than the one for the spike history features. The basis functions are plotted in Figure 3.7.

For this setup we computed the different point estimates and posterior approximations for the weights corresponding to the features describing the spike history dependence (Figure 3.8) as well as for the weights corresponding to the stimulus filters (Figure 3.9). For training, only 2 minutes out of the 20 minutes of recording were used. Another 2 minutes were used for setting hyper-parameters, i.e. prior variances. Given the posterior variances for each of the weights and the basis functions, we can calculate errorbars on the time-course of the coupling and stimulus filters. The filters are defined as the weighted sum of the basis functions. For example, the gamma-functions f_i in equation (3.27) are weighted by the weights, corresponding to the entry in the feature vector $\boldsymbol{\psi}_h$. Errorbars on the coupling filter $f(t)$ can then be estimated using the marginal variances:

$$\begin{aligned} \mathrm{Var}[f(t)\,|D] &= \mathrm{Var}[\mathbf{f}(t)^\top \mathbf{w}\,|D] \\ &= \mathbf{f}(t)^\top \mathrm{Cov}[\mathbf{w}\,|D]\mathbf{f}(t), \end{aligned} \tag{3.28}$$

where $\mathbf{f}(t)$ is a vector of the corresponding basis functions $f_i(t)$ and $\mathrm{Cov}[\mathbf{w}\,|D]$ is part of the posterior covariance matrix corresponding to the weights for the features

described by $f_i(t)$. In the above equation D represents the dataset used for training, containing both, stimulus and spike trains. To illustrate this, we also plotted confidence regions of two standard deviations for the coupling parameters of the population. The confidence intervals for the Gaussian approximation are plotted in red when a Laplacian prior is used and in gray when a Gaussian prior is used. Based on the confidence intervals for the coupling filters, only a few of the connections are actually significant, as can be seen in Figure 3.8. This cannot be concluded from the couplings estimated via MAP or MLE. For example, we see that connections to neuron 1 (first column in Figure 3.8) as well as connections from neuron 1 to any other neuron (first row) are underconstrained by the data, indicated by the large uncertainty for those connections compared to those for others. Consequently, the connections are set to zero by the prior and hence effectively excluded from the model. The strong negative self-feedback coupling, indicating the refractory period can be estimated with a much higher degree of certainty. We also find some significant couplings between neurons, both negatively coupled (e.g. neuron $2 \to 5$) and positively coupled (e.g. $7 \to 2$). The maximum likelihood estimator assigns a non-zero filter to almost every coupling between neurons. The EP mean, however, forces most of the filters to be zero. To quantify the difference in the estimated filters, we calculated the squared difference between the maximum likelihood and the EP-mean weights. This squared difference is 1.5 times larger than the average squared norm of the individual parameter vectors, which indicates that not only the absolute value of the maximum likelihood estimator is larger but also the qualitative shape is different. On the other hand the differences in prediction performance as measured by the likelihood is rather small (see Table 3.2). Thus, close in terms of one quality measure need not necessarily imply close in terms of the other as well. If the posterior uncertainty is small, the parameter vectors are much more constrained by the data and the filters estimated by the maximum likelihood estimator are closer w.r.t. the mean squared distance to the EP-mean. For example this is true for most of the stimulus filters (see Figure 3.9). In contrast, if the posterior uncertainty is rather large, for example for the stimulus filters of neuron 1 and neuron 3, the estimated weights differ more. This suggests, that we do not have sufficient information to estimate *all* parameters, but we are able to extract *some* weights from the given data.

To compare the different estimators quantitatively, we used the same performance measure as for Figure 3.4, namely the negative log likelihood on a test set. To obtain confidence intervals on the performance measure we split the part of the dataset, which was neither used for training nor for validation into 16 different test sets (10% , i.e. 2 minutes for training , 10% for validation and 80% for testing,

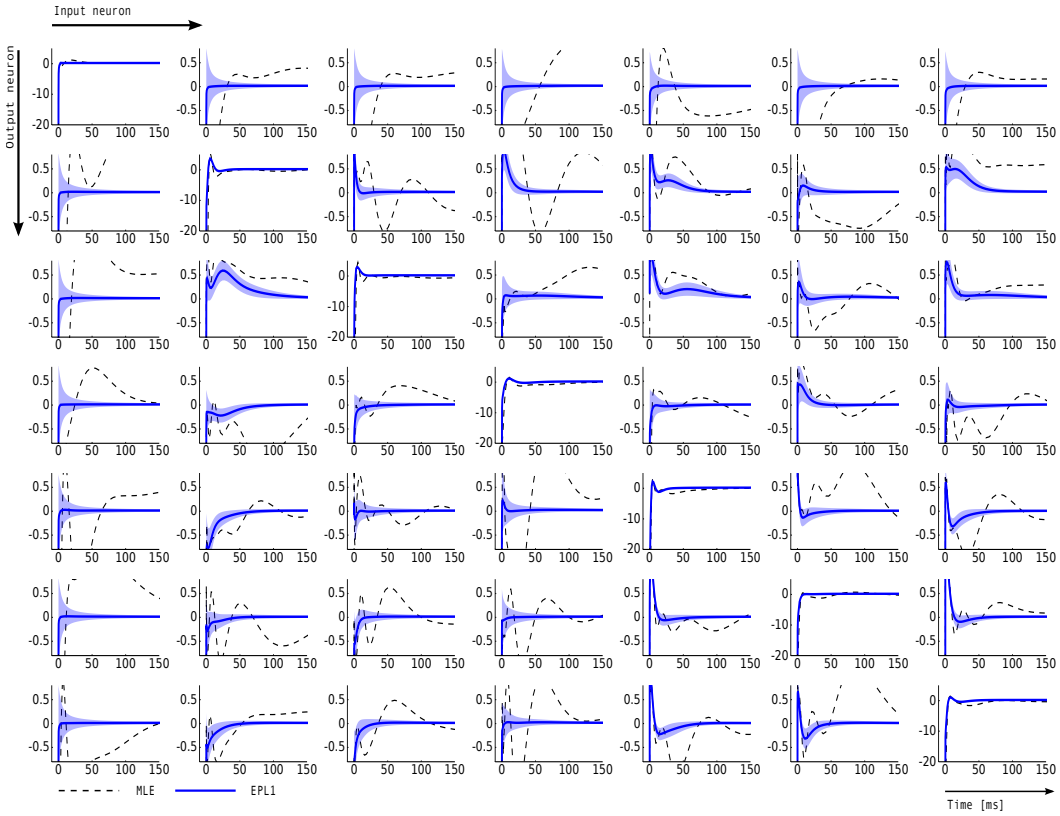

Figure 3.8: Inferred connectivity in the network of seven retinal ganglion cells. Plotted are the induced dependencies by the weights, that is the superposition of basis functions, weighted by the inferred weights from two different estimators: Maximum likelihood (MLE) and approximated posterior when a Laplace prior is used (EPL1). For the EP approximation the posterior mean together with two standard deviations is plotted. Each row corresponds to one output neuron and each column corresponds to a input neuron. Thus, the entry (i,j) describes the influence of a spike of neuron j on the firing rate of neuron i. For example on the diagonal a strong negative coupling on a short time-scale can be observed, representing the refractory period of a neuron. The maximum likelihood estimate as well as thee posterior mean agree on the self-feedback but exhibit a large difference on some couplings, e.g. neuron 1→ 4. In general, neuron 1 seems to be less constrained than other neurons, which is also indicated by the large uncertainty intervals for the connections from and to neuron 1.

split into 16 sets of 1 minute length). The performance values are summarized in Table 3.2. By this performance measure the EP estimate with a Laplcian prior performs significantly better than the MAP estimate with the same prior. The performance difference to the maximum likelihood estimator is not huge, this indicates, that the weights are not sufficiently constrained by one minute slices of the data. Especially the coupling terms not well constrained as can be seen by

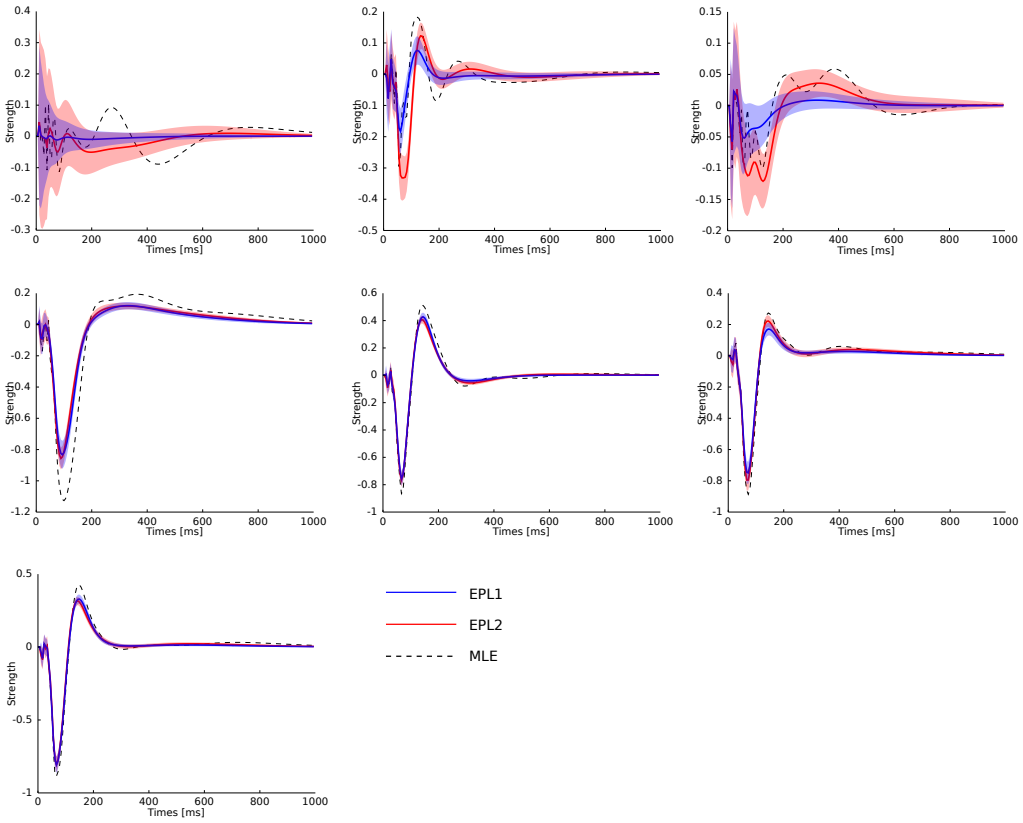

Figure 3.9: Statistical dependence of the neural activity of 7 neurons on the stimulus specified by the superposition of the basis functions plotted in Figure 3.7 weighted by the estimated weights. The same colors for the different estimators as in Figure 3.8 are used. Additionally the posterior mean ($\pm 2\sigma$ confidence intervals) for the EP approximation with a Gaussian prior is plotted in red. Each plot corresponds to one neuron in the same order as in Figure 3.8. As can be seen, the maximum likelihood estimator is overfitting. One sees, that the posterior uncertainty for neuron 1 and also for neuron 3 are much larger as for the other neurons analog to Figure 3.8.

the difference in the estimated filter by the maximum likelihood and the posterior mean, see Figure 3.8. By judging from the data, we do not know if the couplings are needed, hence excluding them from the model, i.e. setting the corresponding weights to zero, seems to be a safe choice. This can be achieved by using a strong prior distribution. The difference between a Gaussian and a Laplace prior is not large for the coupling terms (not shown), for the stimulus filters we see a small difference for the first three neurons, see Figure 3.9. Note, that in cases where there is a significant coupling between neurons, the EP and the maximum likelihood fit agree.

Similarly, we applied the GLM neuron model to another multi-electrode record-

Estimate	neg. log likelihood $\pm 2\sigma$
MLE	$3.609 \cdot 10^{-2} \pm 3.665 \cdot 10^{-4}$
MAPL1	$3.521 \cdot 10^{-2} \pm 2.836 \cdot 10^{-4}$
MAPL2	$3.497 \cdot 10^{-2} \pm 2.592 \cdot 10^{-4}$
EPL1	$\mathbf{3.461 \cdot 10^{-2} \pm 2.459 \cdot 10^{-4}}$
EPL2	$3.716 \cdot 10^{-2} \pm 2.973 \cdot 10^{-4}$

Table 3.2: Mean prediction performance of different point estimates averaged over 16 test sets of 1 minute length. As we do not have access to the true underlying model, the prediction performance here is measured in negative log likelihood score not in differences in likelihoods.

ings of three rabbit retinal ganglion cells. In the previous dataset the stimulus consisted of a full field flicker, hence the receptive fields can be described by a one dimensional curve in time. The stimulus for the rabbit retinal ganglion cells, however, also varied over pixels. Specifically, stimulus consisted of 32767 frames each of which showing a random 16×16 checkerboard pattern with a refresh rate of 50 Hz (data provided by G. Zeck, see [Zeck *et al.* 2005]).

First, in order to investigate the role of the Laplace prior, we trained a single cell GLM neuron model on datasets of different sizes with either a Laplace prior or a Gaussian prior. The models, which have the same number of parameters, were compared by evaluating their negative log-likelihood on an independent test set.

As can be seen in Figure 3.10 the choice of prior becomes less important for large training sets as the weights are sufficiently constrained by the data. For each training set size a separate crossvalidation was carried out. Errorbars were obtained by drawing 100 samples from the posterior.

Fig. 3.11 shows the spatiotemporal receptive field of each neuron, as well as the filters describing the influence of spiking history and input from other cells. For conciseness, we only plot the filters for 80 and 120 ms time lags, but the fitted model included 60 and 140 ms time lags as well. The strongly positive weights on the diagonal of figure 3.11c for the spiking history can be interpreted as "self-excitation". In this way, it is possible to model the bursting behavior exhibited by the cells in our recordings (see also Fig. 3.12). The strongly negative weights at small time lags represent refractory periods. The red lines correspond to 3 standard deviations of the posterior. The first neuron seems to elicit "bursts" at lower frequencies. Note the different scaling of the y-axis for diagonal and off-diagonal terms. By analyzing the coupling terms, we can see that there is significant interaction between cells 2 and 3, but not between any other pair of cells. As our prior assumption is that

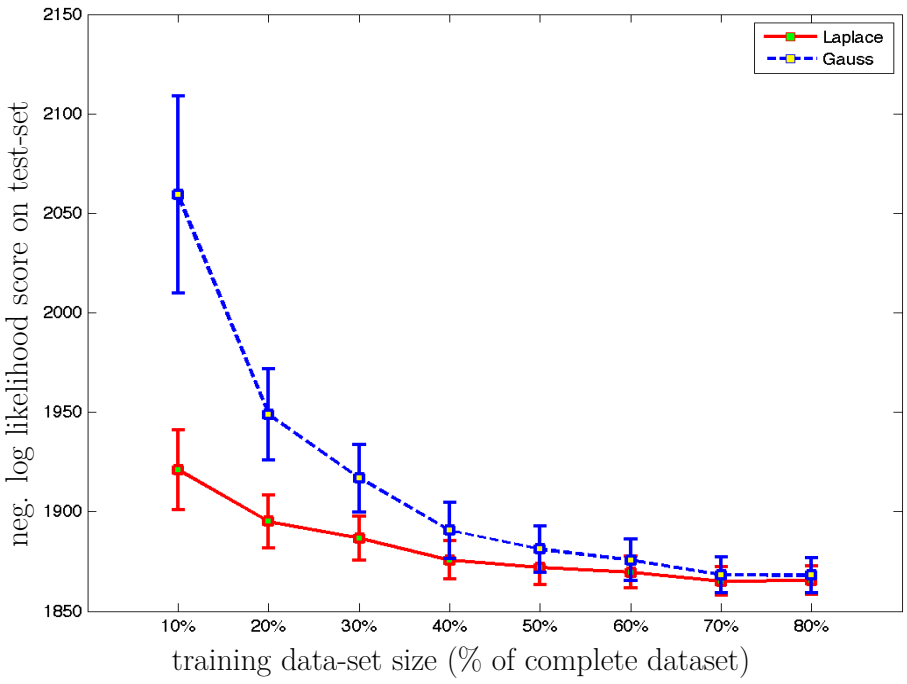

Figure 3.10: Comparison between a Gaussian and Laplacian prior when using different proportions of the available data. When more data is available the prior is less important as the model is well constrained by the data. For the Laplace prior only 20% of the data is needed to achieve the same performance level as for the Gaussian prior with 40 % of the data.

the couplings are 0, this interaction-term is not merely a consequence of our choice of prior. As a result of our crossvalidation it turns out that the prior variance for spike history weights should be set to very large values (ρ= 0.1, variance = $2\frac{1}{\rho^2}$) meaning that these are well determinated by the data. In contrast, prior variances for the stimulus weights should be more strongly biased towards zero ($\rho = 150$).

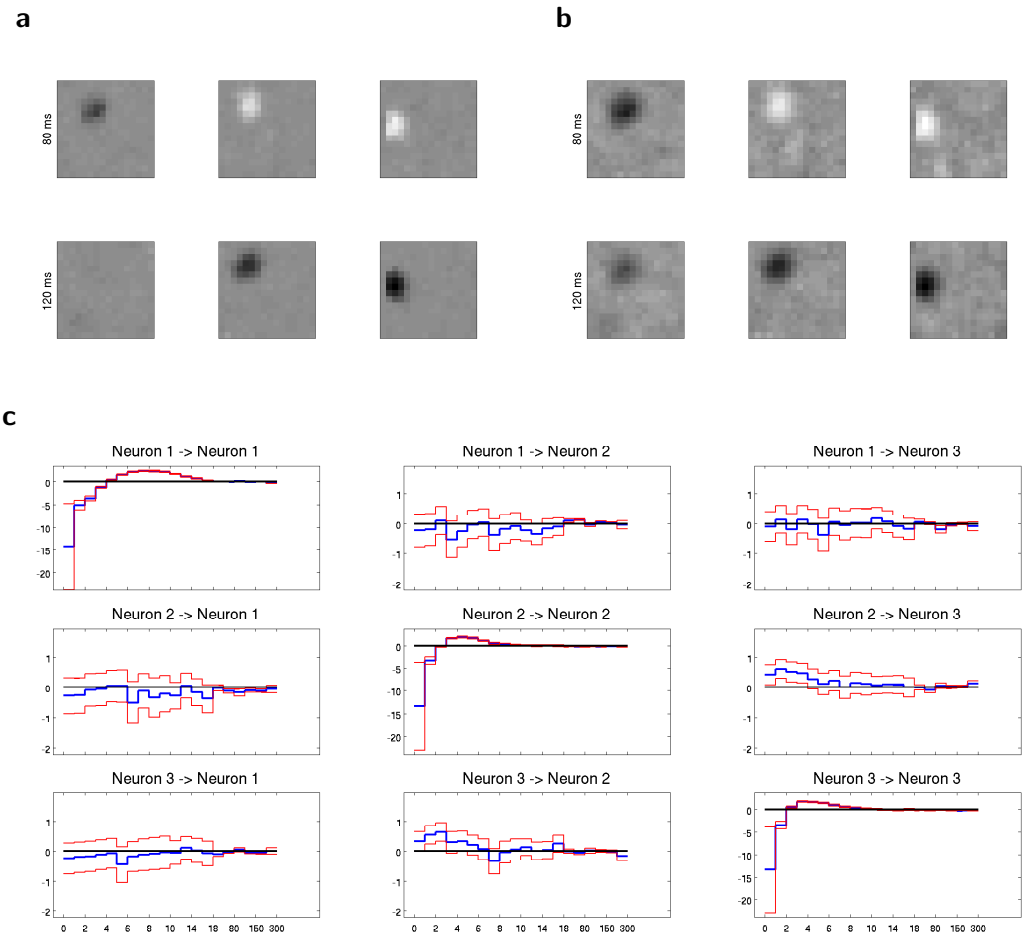

Figure 3.11: (a): Stimulus dependence inferred by the GLM for the three neurons (columns) at different time lags (rows). 2 of 4 time lags are plotted (60, 140 ms not shown). (b): Spike-triggered average for the same neurons and time lags as in (a). (c): Causal dependencies between the three neurons. Each plot shows the value of the linear weight as a function of increasing time lag τ_l (in ms). Shown are posterior mean and three std. dev. (indicated in red). Different scaling of the y-axis is used for diagonal and off-diagonal plots.

Because of the regularization by the prior the spatio-temporal receptive fields are much smoother than the spike-triggered average ones, see Fig. 3.11a. The receptive fields of the STA seems to be more smeared out which might be due to

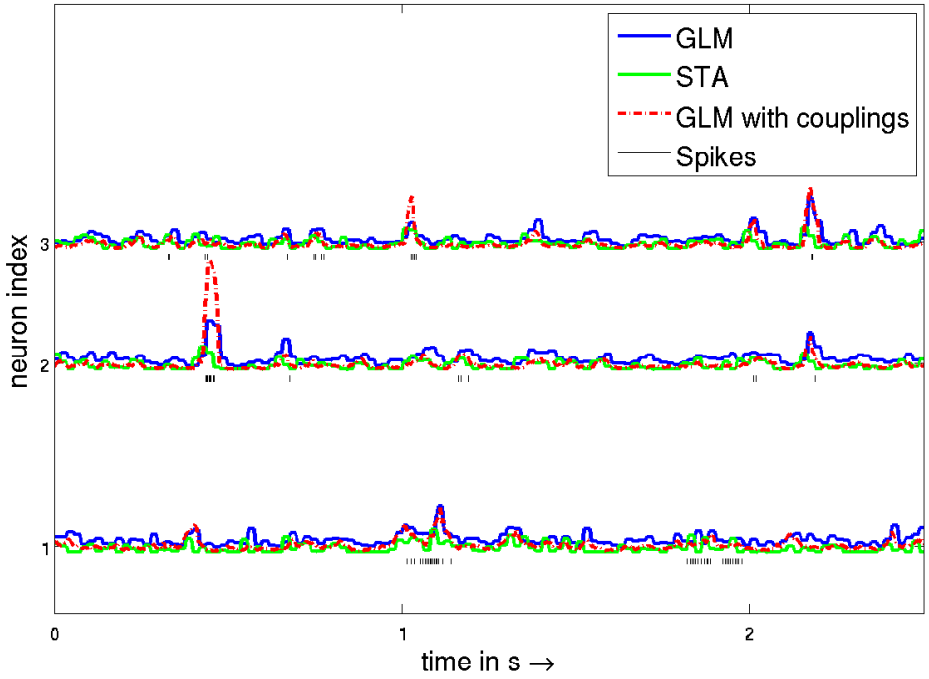

Figure 3.12: Predicted rate for the GLM neuron model with and without any spike history and the predicted rate for the STA for the same neurons as in the other plots. For the STA the linear response is rectified. Rate for the GLM with spike dependence is obtained by averaging over 1000 sampled spike-trains. Rates are rescaled to have the same standard deviation.

	STA	GLM	GLM with couplings
Neuron 1	0.2199	0.2442	0.3576
Neuron 2	0.1746	0.2348	0.3320
Neuron 3	0.1828	0.3319	0.4202
Mean	0.1924	0.2703	0.3699

Table 3.3: Predictions performance of different models. Entries correspond to the correlation coefficient between the predicted rate of each model and spikes on a test set. Both rate and spikes are binned in 5 ms bins. The first GLM models neither connections nor self-feedback.

the fact that it cannot model bursting behavior. The more conservative estimate of the neuron model should increase the prediction performance. To verify this, we calculated the linear response from the spike-triggered average and the rate of our GLM neuron model. In order to have the same number of parameters we neglected all connections. As a model free performance measure we used the correlation

coefficient between the spike trains and the rates (each are binned in 5 ms bins). For the GLM with couplings, rates were estimated by sampling 1000 spike trains with the posterior mean as linear weights. As our model explicitly includes the nonlinearity during fitting, the rate is more sharply peaked around the spikes, see Fig. 3.12. The prediction performance can be increased even further by modeling couplings between neurons as summarized in Tab. 3.3.

3.4.4 Modeling complex cells: How many filters do we need?

Complex cells in primary visual cortex exhibit strongly nonlinear response properties which cannot be well described by a single linear filter, but rather requires a set of filters. A common approach for finding these filters is based on the covariance of the spike-triggered ensemble: Eigenvectors of eigenvalues that are much bigger (or smaller) than the eigenvalues of the whole stimulus ensemble indicate directions in stimulus space to which the cell is sensitive to. Usually, a statistical hypothesis test on the eigenvalue-spectrum is used to decide how many of the eigenvectors e_i are needed to model the cells [Simoncelli *et al.* 2004, Touryan *et al.* 2002, Rust *et al.* 2005, Van Steveninck & Bialek 1988]. Here, we take a different approach: We use the confidence intervals of our GLM neuron model to determine the relevant dimensions within the subspace revealed by STC. We first apply STC to find the space spanned by a set of eigenvectors that is substantially larger than the expected dimensionality of the relevant subspace. Next, we fit a nonlinear function n_i to the filter-outputs $f_i(\mathbf{s}(t)) = \langle \mathbf{s}(t), e_i \rangle$. Finally, we linearly combine the $n_i(t)$, resulting in the following features describing the stimulus:

$$(\boldsymbol{\psi}_s)_i(\mathbf{s}(t)) = n_i(f_i(\mathbf{s}(t))) \tag{3.29}$$

As the model is linear in the weights $\mathbf{w}_i$, we can use the GLM neuron model to fit these weights and obtain confidence intervals. If a filter $f_i(t)$ is not needed for explaining the cells response, its corresponding weight $\mathbf{w}_i$ will automatically be set to zero by the model due to the Laplace prior. This provides an alternative, model-based method of determining the number of filters required to model the cell. The significance of each filter is not determined by a separate hypothesis test on the spectrum of the spike-triggered covariance, but rather by assessing its influence on the neural activity within the full model.

As in the previous application, we can model the spike history effects with an additional feature vector $\boldsymbol{\psi}_h$ to take into account temporal dynamics of single neurons or couplings.

Before applying our method to real data, we tested it on data generated from an

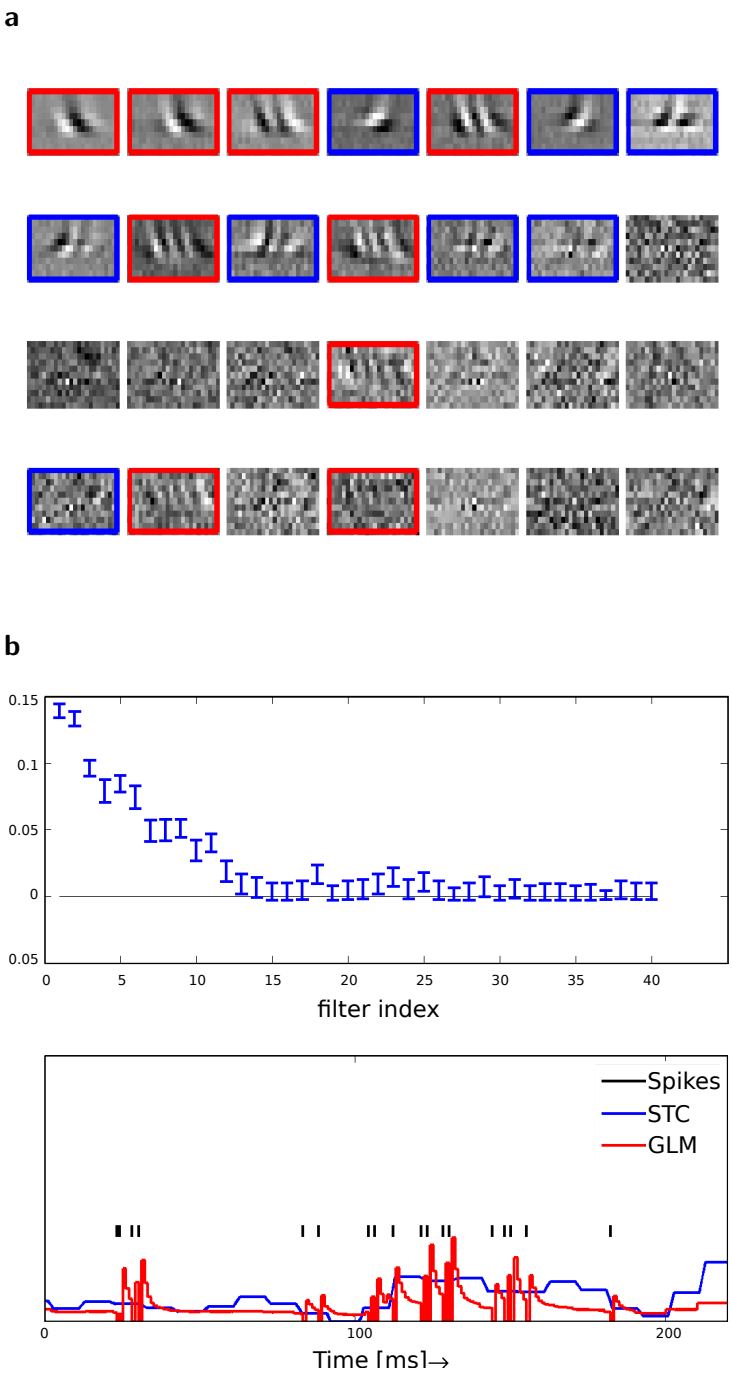

Figure 3.13: (a): 24 out of 40 Filters estimated by STC. The filters are ordered according to their log-ratio of their eigenvalue to the corresponding eigenvalue of the complete stimulus ensemble (from left to right). Highlighted filter are those with significant non-zero weights, red indicating excitatory and blue inhibitory filters. (b) Upper: Posterior mean +/- 3 std. dev. Filter indices are ordered in the same way as in (a). Lower: Predicted rate on a test set for STC and for the GLM neuron model with spike history dependence on a test set.

artificial complex cell similar to the one in [Rust *et al.* 2005]. On this simulated data we were able to recover the original filters. We then fitted this GLM neuron model to data recorded from a complex cell in primary visual cortex of an anesthetized macaque monkey (same data as in [Rust *et al.* 2005]). We first extracted 40 filters which eigenvalues were most different to their corresponding eigenvalues of the complete stimulus ensemble. Any nonlinear regression procedure could be used to fit a nonlinearity to each filter output. We used a simple quadratic regression technique. Having fixed the first nonlinearity we approximated the posterior as above. The resulting confidence intervals for the linear weights are plotted in Fig. 3.13b. The filters with significant non-zero weights are highlighted in Fig. 3.13a. Red indicates exitatory and blue inhibitory effects on the firing rate. Using 3 std. dev. confidence intervals 9 excitatory and 8 inhibitory filters turned out to be significant in our model. The number of filters is similar to that reportet in Rust et al., who regarded 7 excitatory and 7 inhibitory filters as significant [Rust *et al.* 2005]. The rank order of the linear weights is closely related but not identical to the order of eigenvalues, as can be seen in Fig. 3.13b, top.

3.4.5 Approximating other neuron models

We have seen that inference in generalized linear models can be done efficiently. Usually likelihood computations for other neuron models, especially neuron models based on (stochastic) differential equations, imply sampling or solving of integral equations (see [Risken 1989, Paninski *et al.* 2008]). Hence, if we could approximate such a neuron model with a generalized linear model by expanding the stimulus and the spike-history into a suitable feature space, we could use the efficient inference techniques available for the GLM and use them for inference in other neuron models as well. In this section we investigate the approximation ability of a GLM via such a non-linear feature space for the leaky integrate-and-fire model. In general, if we knew the likelihood of another neuron model $p(t|H_t, \mathbf{s}_t)$ for a given spike-history H_t and stimulus $\mathbf{s}(t)$, a simple idea is to discretize the time into small time bins δt and set the intensity function of the GLM to:

$$\lambda(t|H_t, \mathbf{s}(t))\delta t = p(\text{spike} \in [t + \delta t)|H_t, \mathbf{s}(t)) \tag{3.30}$$

For the special case of the leaky integrate and fire neuron model, the idea can be further simplified by approximating the likelihood with a hazard function which only depends on the noise-free solution of the membrane potential [Plesser & Gerstner 2000, Koyama & Paninski 2009]. In [Plesser & Gerstner 2000]

various choices of such hazard functions are discussed. As the hazard function describes the risk of an escape across the threshold of the membrane potential, such an approximation is also called escape rate approximation. However, all the proposed approximations assume, that the receptive field as well as the noise-level and the leak is known. Here, we investigate how well such an approximation based on the GLM hazard function can predict spikes from a LIF when the receptive field has to be inferred from the data.

Recall the basic model for a (single) leaky integrate-and-fire neuron (see also section 2.1 and 4.2.1) with receptive field $\mathbf{r}$:

$$\begin{aligned} \mathrm{d}V_t &= (\mathbf{r} \star \mathbf{s}(t) - \tau(V_t - V_r))\,\mathrm{d}t + \sigma \mathrm{d}B_t \\ V_{t^+} &= V_r \qquad \text{if } V_{t^-} = \theta \end{aligned} \tag{3.31}$$

If we represent the receptive field $\mathbf{r}(t)$ as a superposition of several basis functions $f_i(t), i = 1, \ldots, M$ with corresponding coefficients $\mathbf{R}_i$, we obtain:

$$\mathrm{d}V_t = \left(\sum_i \mathbf{R}_i f_i \star \mathbf{s}(t) - \tau(V_t - V_r) \right) \mathrm{d}t + \sigma \mathrm{d}B_t \tag{3.32}$$

If we now split the membrane potential into individual components corresponding to the basis functions and for simplicity assume the reset potential V_r to be zero, analogous to equation (2.2), we obtain:

$$\begin{aligned} \mathrm{d}V_t^i &= \left(f_i \star \mathbf{s}(t) - \tau V_t^i \right) \mathrm{d}t \\ \Rightarrow V_t^i &= \exp(-\tau(t - t_-)) \int_{t_-}^{t} \exp(\tau(\xi - t_-)) f_i \star \mathbf{s}(\xi) \mathrm{d}\xi \\ \Rightarrow \mathbf{V}_t &= \sum_i \mathbf{R}_i V_t^i + U_t, \end{aligned} \tag{3.33}$$

where t_- is the time of the last spike and U_t is a Gaussian noise term with zero mean and a variance which is evolving according to equation (2.4). Note, that the noise process U_t is temporally correlated due to the leaky integration of the Brownian motion term. Thus, the mean membrane potential $\mathbf{V}_t$ can be written as a linear superposition of time varying features with fixed weightings $\mathbf{R}$. The generation of spike times is then governed by the additional noise U_t and the threshold. To mimic the reset/renewal property of the LIF neuron within the class of GLMs, we can set $(\boldsymbol{\psi}_{h,s})_i(t) := V_t^i$ including the reset to zero after each spike. We have

indexed the feature vector $\boldsymbol{\psi}$ with h and s to emphasize, that it depends on both the spiking history and the stimulus. In general, the leak term τ has to be estimated as well. For the sake of simplicity, however, we assume in the following that τ is known[2]. The noise term U_t can equivalently be seen as a temporal modulation of the threshold. As spikes in a GLM are generated according to the instantaneous intensity $f(\boldsymbol{\psi}_{h,s}(t)^\top \mathbf{R}) \approx f(\mathbf{V}_t)$, it can be interpreted as a soft threshold spike-generation. If we set the noise σ to zero and use $f(x) = c \cdot \mathbb{1}_{x>\theta}, c \gg 1$ as non-linearity, we obtain an almost deterministic spike-generation which is close to the deterministic LIF model. Thus, we expect the GLM with such a feature space to give a reasonable approximation to the leaky integrate-and-fire model for sufficiently small noise and steep non-linearity. In order to further adjust the probability of a threshold crossing to the time-varying variance of the noise term, we could also add another ($(n+1)$-th) feature evolving from the last spike according to:

$$\begin{aligned} (\boldsymbol{\psi}_h(t))_{n+1} &= \log\left(1 - \int_{-\infty}^{\theta} \mathcal{N}(x|0, \sigma(t))\mathrm{d}x\right) \\ \sigma(t) &:= \frac{\sigma^2}{2\tau}\left(1 - \exp\left(-2\tau(t - t_-)\right)\right) \end{aligned} \tag{3.34}$$

where t_- is the time of the last spike and σ is the noise level of the original leaky integrator. Thus, $(\boldsymbol{\psi}_h(t))_{n+1}$ reflects the log probability of the membrane potential being above threshold in time bin t given that the last spike was at t_-. Note, that this is only an approximation as we do not make use of the fact, that the membrane potential has not crossed the threshold in the time bins between t_- and t. This features depends only on the time of the very last spike and hence also reflects the renewal property of the leaky integrator.

To investigate the ability of a GLM to approximate a LIF neuron when using custom made features, we generated spike trains from a LIF and calculated the EP approximation to the posterior as in the previous sections. Specifically, we simulated two different LIFs:

High pass: LIF with a Gabor like receptive field, see Figure 3.14b.

Low pass LIF with a Gaussian receptive field, see Figure 3.14a.

Due to the receptive fields, spikes from the LIF with the Gabor receptive field are more irregular than the ones generated with the Gaussian receptive field, see Figure 3.15. For each of these two LIFs we generated 100 seconds of a white noise

[2]Generally, τ could be set via a crossvalidation procedure as it plays the role of a hyperparameter here.

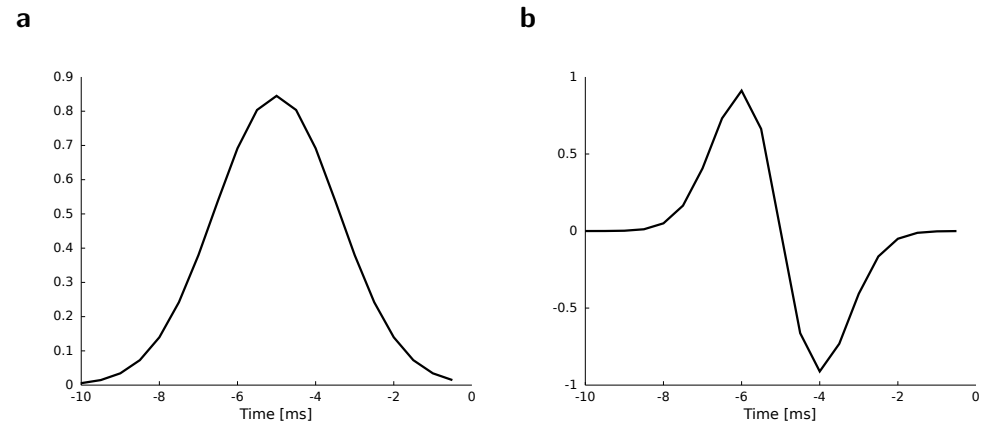

Figure 3.14: Receptive fields for the LIF used to generate spikes from a LIF. Each receptive field is discretized into 20 bins of 0.5 ms length. **Left:** Gaussian receptive field, corresponding to a low pass filter. **Right:** Gabor receptive field, corresponding to a high pass filter.

stimulus with a refresh rate of 0.5 ms. Other parameters of the LIF were fixed to $\sigma = 0.5$ [mV], $\tau^{-1} = 20$ [ms]. The threshold for the LIF with the low pass receptive field was set to 12, whereas the threshold for the high pass case was set to 4, in order to obtain roughly the same rates in both cases. The spike trains obtained, were fitted with three GLMs with different feature spaces:

LNP: GLM with 20 features describing the raw stimulus in the last 20 time-bins.

renewal GLM: GLM with 20 features defined as in equation (3.33). The basis functions for equation (3.33) were set to indicator functions $f_i(t) = \mathbb{1}_{(t-(i+1)0.5, t-i0.5]}$ representing the same set of stimulus time-bins as the previous LNP.

LNP + GLM: GLM containing features describing the raw stimulus as well as features from the renewal GLM. Furthermore we added the variance feature of equation (3.34).

Note, that the dimensionality of the first two GLMs are the same, whereas the dimensionality of the last GLM is twice as large. To analyze the effect of custom made feature spaces including the reset property, we generated spike trains for each of these fitted models and compared them to the ones generated from the original LIF, see Figure 3.15. To sample those spike trains we first sampled a weight vector according to the posterior distribution. Given the sampled weights we then generated the spike trains from the corresponding GLM. In this way we can draw

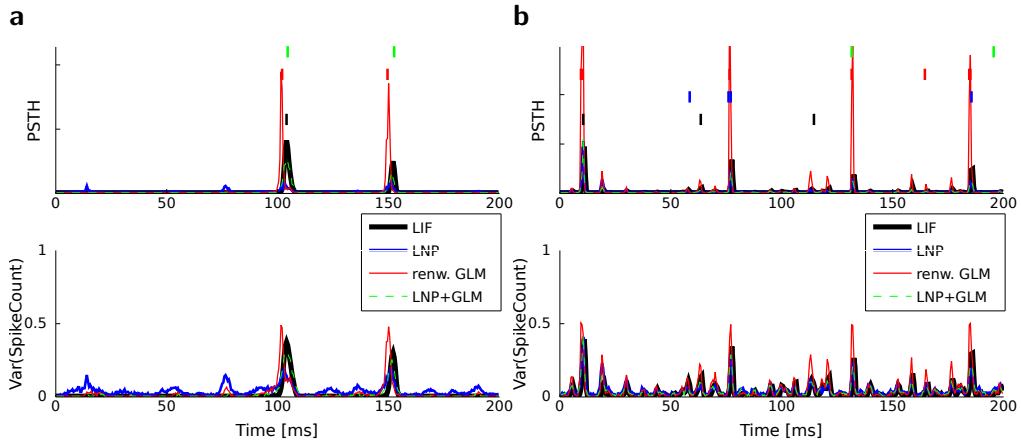

Figure 3.15: Comparison of mean and variance of a PSTH for the different GLM approximations to the LIF. In the top two panels the mean spike count is plotted for the different approximating GLMs: LNP, renewal GLM, LNP+GLM and black for the LIF. In the bottom two panels the variance of the spike count is plotted. The left two panels correspond to the low-pass setting whereas the two right panels show the results for the high-pass setting. In addition sample spike trains (vertical bars) are plotted in the top two panels with the same colors as used for the PSTHs. Both GLMs including features for the reset property and the variance feature capture the sharp onset for the spike counts as indicated by the variance of the spike counts. Due to the nonlinearity used, the onset of the PSTH of the renewal GLM is slightly earlier than the others (see text).

samples from the predictive distribution. To illustrate the predictive distribution, we repeated the simulations 10000 times and calculated the peri stimulus histograms (PSTHs) for the different models, see Figure 3.15.

To quantify the prediction performance, we calculated the average log-loss of the predictive distributions for the different GLMs in Table 3.4. Specifically, the average log-loss is given by (see also equation (3.11)):

$$\langle -\log p(\{t_i\}|\mathrm{GLM})\rangle = \left\langle -\log \int p(\{t_i\}|\mathbf{w},\mathrm{GLM})p(\mathbf{w}|D,\mathrm{GLM})\mathrm{d}\mathbf{w}\right\rangle, \qquad (3.35)$$

where D is data set used for inference and the average $\langle\rangle$ has to be taken over spike trains $\{t_i\}$, generated from the original LIF. As the integral cannot be performed analytically, we estimated the average log-loss by sampling weights from the posterior and spike trains from the LIF. In Table 3.4 we see, that the performance of the predictive distribution for the GLM including only the renewal features is significantly better than the GLM including only stimulus features. In terms of

	low pass LIF	high pass LIF
LNP	122.03 ± 0.69	170.85 ± 0.6
renewal GLM	**76.63 ± 0.44**	**162.46 ± 0.56**
LNP+GLM	**75.47 ±0.44**	168.39 ± 2.72
Constant rate	161.844 ± 0.8	231.32 ± 0.82

Table 3.4: Prediction performance of different GLMs when using different feature spaces. The prediction performance is measured in average log-loss of the predictive distributions of the different GLMs. In the first row the prediction performance for the GLM using only stimulus features is shown. The second row shows the average log-likelihood for the GLM which only uses the renewal features. At least for the high-pass case, the performance for the GLM which combines all features and has the additional variance feature as shown in the third row does not improve the prediction performance significantly. As a reference we also show the average log-likelihood for the constant rate estimator in the last row.

average log-loss the more complex model including both types of features and the additional variance feature (equation (3.34)) does not improve the performance substantially, at least in the high pass case. However, in terms of predicted PSTH using both types of features leads to a better match between the predicted PSTH and the PSTH from the original LIF. Furthermore, when sampling from the renewal GLM the onset of the resulting PSTH is slightly earlier than the PSTH from the LIF, see Figure 3.15. As we used an exponential non-linearity the GLM sometimes predicts spikes, when the membrane potential has not reached the threshold yet. Due to the reset of the membrane potential this results in a small latency shift in the PSTH.

3.5 Discussion

Bayesian inference methods are particularly useful for system identification tasks where a large number of parameters need to be estimated. By specifying a prior over the parameters a full probabilistic model is obtained that provides a principled framework for regularizing the model complexity. Furthermore, knowledge of the posterior distribution allows one both to derive point estimators that are optimized for loss functions that are suitable to the problem at hand and to quantify the uncertainty about such estimates.

A major hurdle for using a Bayesian approach is that computing the posterior distribution is often intractable. Even for numerical approximation techniques of the posterior distribution there is usually — *a priori* — no guarantee how well they work. Therefore, it is important to perform careful quality control studies if such

methods are to be applied to a new estimation problem. In this chapter, we presented such control studies for approximate Bayesian inference in the generalized linear models of spiking neurons using Expectation Propagation (EP) and compared it to standard methods like maximum likelihood and MAP estimates. Expectation Propagation provides both a posterior mean and a posterior covariance approximation. These first and second-order moments are sufficient to obtain a rough sketch of the location and dispersion of the posterior distribution. The posterior mean, in particular, can be used as a point estimator which is known to minimize the mean squared error loss. This loss function is an expedient choice if one aims at reconstructing the filter shapes. As we have shown in this work, the posterior mean estimate obtained with EP yields a smaller mean squared reconstruction error of the parameters than maximum likelihood or MAP estimation.

It should be noted, however, that the filter shapes represent statistical couplings only. Clearly, the existence of a statistical coupling does not necessarily imply the existence of a physical coupling as well. Statistical dependence could, for example, also be a consequence of common input, or other indirect couplings. In fact, it is known that noise correlations between retinal ganglion cells are mainly due to common input, and not direct synaptic couplings [Trong & Rieke 2008]. In the model an inferred coupling simply indicates that there is a dependence between the neurons which cannot be explained by the stimulus filters or the neural self-couplings.

Receptive field estimation aims at a functional characterization of neural response properties. Therefore, it is natural to compare different estimates by asking how well they can predict spike trains generated in response to new test data. Evaluating the performance of predicting a particular spike train usually involves the use of a spike train metric [Victor & Purpura 1997], as the predicted spike trains have to be compared to the observed spike trains. In general, one wants to compare models, and not only particular spike trains, and therefore averages the prediction performance across very many samples from the two models one wants to compare.

The Bayesian framework offers a principled way to obtain an optimal point estimate which minimizes the loss function averaged across the posterior distribution. Although it is unlikely that this optimization problem can be solved analytically, one can sample weights from the posterior and then sample several spike trains for these given weights. In other words we can generate samples form the predictive distribution. For the prediction performance measure specified by the loss in equation 3.11, for example, an optimal point estimate would be given by those weights which on average yield the largest likelihood for the ensemble of spike trains drawn from the predictive distribution. Neither the MAP nor the posterior mean

is optimal with respect to this criterion. Theoretically, the MAP is optimized for the zero-one-loss, whereas the posterior mean is optimized for the squared error loss [Lehmann & Casella 1998]. In Appendix A.2, we demonstrate on a simple, concrete example (estimation of the probability of a coin flip and log-loss as loss function) that an optimized predictor will perform better (on average) than the MAP-estimate, irrespective of what data was observed. Clearly, this approach is only possible if one has at least an approximate model of the posterior, as we have presented here.

For a single GLM this will yield a set of parameters, which are guaranteed to be optimal on average. The optimality of course only holds if the model is correct (i.e. the observed spike trains are indeed samples from a GLM), the prior is appropriately chosen, and the posterior distribution can be calculated precisely. In practice, it is not clear how justifiable each of the the three assumptions is going to be. Therefore, it is an interesting open question of how much better point-estimates which are optimized using this approach will perform when compared to other optimization methods. Empirically, we observed that the posterior mean estimate obtained with EP is always better then the MAP with respect to squared error loss. With respect to the prediction error, the MAP performed slightly better than the EP posterior mean estimate if the weights were drawn from a Gaussian or Laplacian distribution, while the EP posterior mean was better than the MAP estimator if the weights were drawn from the truly sparse distribution. Of course one could also directly use the predictive distribution as it will in general assign higher likelihood to unseen spikes than any point estimate. However, the predictive distribution cannot be described by a single GLM as it is an average over many models.

Our study also provides some insights about the effect of different kinds of prior distributions on the estimation performance. The choice of prior in the Bayesian framework offers a principled way of regularization. Here, we compared specifically a Gaussian and a Laplacian prior. While there was almost no difference in performance between the EP posterior mean estimator for the Laplacian and the Gaussian prior if the true prior was Gaussian or Laplace, the assumption of a Laplacian prior led to a substantial advantage when the true weight vectors had only a few non-zero components. This confirms the intuition that one can profit from using a Laplacian prior if one sets up a large number of candidate features of which only a few are likely to be useful in the end. Interestingly, for the MAP estimator, the use of a Laplacian prior almost always led to a substantial impairment and resulted in a relatively small improvement only w.r.t the prediction performance if the weights were sampled from a sparse distribution for which almost all coefficients are zero.

While the posterior mean, and even more so the MAP estimator can strongly depend on the particular choice of prior distribution, this indeterminacy is a problem only if the dispersion of the posterior distribution is not taken into account appropriately. This is a strong case for the use of EP as the MAP estimator does not provide any control to what extent the result is actually constrained by the data. By also computing the posterior covariance rather than just a point estimator, we obtain confidence intervals which can serve exactly to this purpose. For the retinal ganglion cell data analyzed in section 3.4.3, for example, it allowed us to distinguish between neuronal couplings, that are significant and others which were not (see neuron 1 in Figure 3.8). Also, in the context of spike-triggered covariance analysis, we used our method to determine the relevant stimulus subspace within the space spanned by the eigenvectors. Our subspace selection method is directly linked to an explicit neuron model which also takes into account the spike-history dependence of the spike generation. Whenever the confidence intervals were large, the maximum likelihood estimator deviated substantially from the Bayesian point estimators, hence indicating overfitting.

4

Decoding with leaky integrate-and-fire neurons

4.1 Introduction

In the previous chapter we have considered the encoding problem. There, we were aiming at predicting spikes given a particular sequence of stimuli. The nervous system on the other hand has to solve at least implicitly the inverse problem. That is, given an observed spike-train, what stimulus is likely to have produced this particular neural response. Understanding how stimuli and other inputs to neurons can be decoded from their spike patterns is an essential step towards understanding neural codes.

If the encoding mapping was one-to-one this would be an easy task. However, due to internal noise and common input, predicting a stimulus given a spike train is a nontrivial problem. As a first approximation, a similar technique as the spike-triggered average can be used. That is, the optimal linear decoder can be calculated [Bialek *et al.* 1991]. For orientation tuning, another popular method is the population vector method which reconstructs the stimulus by a weighted superposition of the preferred orientations [Georgopoulos *et al.* 1982].

Instead of searching for an explicit decoder a commonly studied ques-

tion is: what is the best reconstruction any decoder could possibly achieve, given an explicit encoding model. Along these lines Fisher Information is a widely used tool for accessing the quality of an encoding scheme [Paradiso 1988, Seung & Sompolinsky 1993, Abbott & Dayan 1999]. Another approach is to ask how well any two given stimuli can be discriminated on the bases of the neural responses [Shadlen *et al.* 1996, Berens *et al.* 2009].

Most of the existing studies have focused on static encoding models based on spike counts only. Many sensory inputs, however, change continuously in time and have variations across a large range of different time scales. Similarly, the occurrence of spikes can depend on continuous electrophysiological signals such as local field potentials [Montemurro *et al.* 2008, Rasch *et al.* 2008]. In this chapter, we seek to achieve a better understanding of how such continuous signals can be decoded from neuronal spike trains, and how the basic biophysical dynamics of individual neurons affect the encoding process.

We will investigate these questions using leaky integrate-and-fire neurons (LIFs) [Stein 1967, Tuckwell 1988]. Leaky integrators constitute a natural choice as they capture basic dynamical properties of neurons, yet are still amenable to analytical studies of dynamic encoding. In this model, a spike is emitted as soon as the integrated input reaches a threshold. Thus, the relative timing of spikes will contain information about the stimulus in the recent past. In the noiseless case, an elegant solution has been proposed for decoding a time-varying stimulus from a population of integrate-and-fire neurons based on computing the pseudo-inverse [Seydnejad & Kitney 2001].

Here, we seek to generalize from the noiseless to the noisy case. Specifically, we study decoding rules for reconstructing time-varying, continuous stimuli from populations of leaky integrate-and-fire neurons with noisy membrane thresholds. Incorporating noise into the model does not only make the model more realistic, but also naturally leads to a Bayesian approach to population coding [Rao *et al.* 2002, Huys *et al.* 2007, Natarajan *et al.* 2008]. Each spike constitutes a noisy measurement of the underlying membrane potential and, using the Bayesian formalism, this relationship can be inverted in order to infer the posterior distribution over stimuli [Lewi *et al.* 2008, Paninski *et al.* 2007]. While many studies have addressed Bayesian population codes and the representation of uncertainty in neural populations [Pouget *et al.* 2000, Rao *et al.* 2002, Rao 2005, Ma *et al.* 2006], the question of how posterior distributions can be decoded from the spike-times of LIFs has not been studied in detail. Natarajan and Huys [Huys *et al.* 2007, Natarajan *et al.* 2008] analyzed probabilistic decoding of continuously varying stimuli, but they did not use the LIF neuron model but an inhomogeneous Poisson point

process.

A Bayesian decoding rule does not only return a point estimate of the stimulus, but also an estimate of the posterior covariance, representing the residual uncertainty about the stimulus. This uncertainty estimate is of critical importance for a "spike-by-spike" decoding scheme [Wiener & Richmond 2003], as it allows one to appropriately weight each observation by its reliability. In addition, the uncertainty directly relates to the accuracy of the neural code. By inspecting the posterior variance of different stimulus features, one can gain insight into the accuracy with which different features are represented by the population.

For the sake of clarity, we choose a simple threshold noise model, which does not affect the dynamics of the integration process but only sets the threshold to a new random value whenever a spike has been elicited [Gerstner & Kistler 2002]. The generation of spikes in this model class can be described by a renewal process, see also chapter 2. A Gamma point process is obtained as special case in the limit of a large membrane time constant when the threshold values are drawn from a Gamma distribution. In particular, when the exponential distribution is chosen, the spike generation process constitutes an inhomogeneous Poisson process. The Gamma distribution is a computationally convenient distribution which ensures positiveness of the threshold. Therefore, this choice of noise model is conceptually simple, but nevertheless can be used to model a wide range of different spiking statistics. However, even for this simple noise model, the exact shape of the posterior distribution over stimuli can not be obtained in closed form in general and approximations have to be used. Here, we derive three decoding rules based on Gaussian approximations to the posterior distribution. We show that the simple decoder which originates from the noiseless case is biased when introducing threshold noise. We then derive an expression for the bias length and state conditions under which this leads to an improved estimator of the stimulus. Furthermore, we show how this estimate can be updated iteratively every time a new spike is observed.

The chapter is organized as follows: In section 4.2 we describe the basic encoding model as well as the stochastic description of the time-varying input. The decoding in the noiseless case can be extended to include threshold noise as well. This leads to an approximate likelihood, from which we derive several approximations to the full posterior distribution in section 4.3. In section 4.4 we compare the resulting Bayesian decoding schemes to alternative reconstructions, such as the linear decoding [Bialek *et al.* 1991] and the Laplace approximation [Paninski *et al.* 2007, MacKay 2003, Rasmussen & Williams 2006] based on the likelihood approximation. Finally, in section 4.5, we apply the decoding schemes to different scenarios which illustrate different aspects of neural population coding.

4.2 Encoding

The encoding process is split up into two parts: The first one is the neural encoding part, which characterizes the spike generation process for a given stimulus. The second part describes the stimulus ensemble.

4.2.1 Leaky integrate-and-fire neuron with threshold noise

We start with the classic leaky integrate-and-fire neuron model [Tuckwell 1988, Gerstner & Kistler 2002]. It consists of a membrane potential $\mathbf{V}_t$ which accumulates the effective input $\mathbf{I}_t$. Here, $\mathbf{V}_t$ and $\mathbf{I}_t$ are scalar functions if a single neuron is modeled, or vectors if a population is considered. Whenever the membrane potential of a neuron n reaches a pre-specified threshold θ^n a spike is fired and the membrane potential is reset to zero, i. e. $\lim_{\varepsilon\to 0}(\mathbf{V}_{t_k+\varepsilon})_n = 0$. In addition to the input $\mathbf{I}$, there is a leak term which drives the membrane potential back to zero when no input is present. Correspondingly, the sub-threshold dynamics of the membrane potential can be described by the following ordinary differential equation (ODE):

$$\tau \mathrm{d}\mathbf{V}_t = \mathbf{I}_t \mathrm{d}t - \mathbf{V}_t \mathrm{d}t. \tag{4.1}$$

The time constant τ specifies the time scale of the neural dynamics. Assuming the time of the last spike is t_{k^-}, the membrane potential at any time t before the next spike is given by

$$\mathbf{V}_t = \exp\left(-\frac{1}{\tau}\left(t - t_{k^-}\right)\right)\frac{1}{\tau}\int_{t_{k^-}}^{t} \exp\left(\frac{1}{\tau}\left(s - t_{k^-}\right)\right)\mathbf{I}_s \,\mathrm{d}s =: F_{[t_{k^-},t)}(\mathbf{I}). \tag{4.2}$$

$F_{[t_{k^-},t)}(\mathbf{I})$ is a linear functional of the stimulus $\mathbf{I}$ depending on the time of the last spike t_{k^-} and the current time point t. Due to the additional spiking nonlinearity that governs the dynamics when the membrane potential reaches the threshold, the LIF neuron performs a complex mapping of continuous signals to spike patterns. A simple way of incorporating noise into our model is to vary the threshold from spike to spike in a stochastic fashion. Every time a spike is fired, the threshold is drawn from a known distribution with density p_θ. Thus for every given (constant) stimulus, the resulting point process is a renewal process.

With these assumptions we can write down the likelihood of observing a spike

train of one neuron for a given stimulus $\mathbf{I}_t$:

$$\begin{aligned} p(t_0, t_1, \ldots, t_n|\mathbf{I}_t) &= p(t_0|\mathbf{I}_{(0,t_0)}) \prod_{k=1}^{n} p(t_k|t_{k-1}, \mathbf{I}_{(t_{k-1},t_k)}) \\ &\approx p(t_0|\mathbf{I}_{(0,t_0)}) \prod_{k=1}^{n} p_\theta(F_{[t_{k-1},t_k)}(\mathbf{I})|t_{k-1}, \mathbf{I}_{(t_{k-1},t_k)}) \left| \frac{\mathrm{d}F_{[t_{k-1},t_k)}(\mathbf{I})}{\mathrm{d}t_k} \right|, \end{aligned} \tag{4.3}$$

with $F_{[t_k,t_{k-1})}$ defined as in equation (4.2) and $\mathbf{I}_{(t_{k-1},t_k)}$ denotes the stimulus between t_{k-1} and t_k. The first equality holds because of the renewal property of the spike generation process. In other words, the time of the next spike only depends on the time of the previous spike and the stimulus since then. Subsequently, we change variables from t_k to $F_{[t_{k-1},t_k)}(\mathbf{I})$. Note that $F_{[t_{k-1},t_k)}(\mathbf{I})$ is only a function of t_k because we condition on t_{k-1} and $\mathbf{I}$. As the value of the linear functional at the time of a spike equals the threshold θ, we plug in the density for the threshold p_θ. The change of variables t_k to $F_{[t_{k-1},t_k)}(\mathbf{I})$ is only one-to-one, if one uses the fact, that t_k is the first time $F_{[t_{k-1},t_k)}(\mathbf{I})$ equals the threshold. Therefore, plugging in the threshold distribution without accounting for the problem, that $\mathbf{F}(\mathbf{I})$ may have been super-threshold turns the last equation into an approximation. If we consider a whole population, the likelihood reads:

$$\begin{aligned} p(t_0, t_1, \ldots, t_n|\mathbf{I}_t) &= p(t_0|\mathbf{I}_{(0,t_0)}) \prod_{k=1}^{n} p(t_k|t_{k^-}, \mathbf{I}_{(t_{k^-},t_k)}) \\ &\approx p(t_0|\mathbf{I}_{(0,t_0)}) \prod_{k=1}^{n} p_\theta(F_{[t_{k^-},t_k)}(\mathbf{I})|t_{k^-}, \mathbf{I}_{(t_{k^-},t_k)}) \left| \frac{\mathrm{d}F_{[t_{k^-},t_k)}(\mathbf{I})}{\mathrm{d}t_k} \right|, \end{aligned} \tag{4.4}$$

where t_{k^-} denotes the time of the previous spike of the neuron, which fired a spike at time t_k. The threshold distribution p_θ might be different for different neurons. For notational simplicity, however, we do not indicate this. In the following the spike times t_k are ordered and indexed by the subscript k. Which neuron fired the spike t_k only enters the calculation in the computation of the linear functionals $\mathbf{F}_{t_{k^-},t_k}(\mathbf{I})$. Therefore we drop the dependency of the neuron.

There is no simple way how the sub-threshold condition can be incorporated. However, we can include the condition that at the time of reaching the threshold, the membrane potential $\mathbf{V}_t$ must be increasing by adding the requirement $\frac{\mathrm{d}F_{[t_k,t_{k^-})}(\mathbf{I})}{\mathrm{d}t_k} > 0$ [Pillow & Simoncelli 2002, Arcas & Fairhall 2003].

For the threshold noise we assume a Gamma distribution with shape parameter

α and scale parameter β:

$$p_\theta(\theta) = \theta^{\alpha-1} \frac{e^{-\theta/\beta}}{\beta^\alpha \, \Gamma(\alpha)} \tag{4.5}$$

As a special case, if the input is non-negative and if the time constant goes to infinity, the resulting point process is an inhomogeneous Gamma-renewal process.

In this way we obtain an approximate likelihood, when the threshold is varied at the time of spikes. The case of white input noise and fixed threshold is described in [Paninski *et al.* 2004]. This can equivalently be seen as varying the thresholds continuously according to an Ornstein Uhlenbeck process. For the case of soft-threshold based likelihoods from the family of Generalized Linear Models, see [Jolivet *et al.* 2006, Paninski *et al.* 2007].

4.2.2 Specifying the prior: A model for the stimulus

The prior distribution specifies the assumption about the range and relative frequency of different stimuli. A common approach is to use a maximum entropy prior. In particular, the normal distribution is a maximum entropy distribution for given mean and covariance. As stimuli are functions of time, we have to specify a distribution over functions. We choose a finite set of basis functions $\{f_i\}$ and then specify a distribution over the coefficients from which all possible functions are generated by a linear superposition:

$$\mathbf{I}_t = \sum_{i=1}^{M} \mathbf{c}_i f_i(t). \tag{4.6}$$

The coefficients $\mathbf{c}_i$ are drawn from the Gaussian prior distribution. We denote the mean and the covariance matrix by μ_c and Σ_c , respectively. For stationary processes, a natural choice of basis functions is the Fourier basis. Any superposition of such basis functions will result in a smooth function. Defining a covariance structure for the coefficients directly translates into the structure of the power-spectrum. Thus, $\mathbf{I}_t$ is a finite-dimensional Gaussian process. Using a finite number of basis functions poses a potential difficulty for the spike generation process described in the previous section. If one uses basis functions which are bounded, so will be any sample from the input process. Therefore, there is a non-zero probability that a threshold is drawn which could never be reached by the membrane potential. However, if we use a flat power-spectrum, i. e. isotropic covariance for the coefficients, and increase the number of Fourier basis functions the process will converge to a Brownian motion. For Brownian motion as input, the membrane potential is

an Ornstein-Uhlenbeck process and therefore will eventually exceed any threshold. For the simulations in this chapter, we never observed an infinitely long inter-spike interval.

Using this model for the stimulus we can rewrite the linear functional of the stimulus as an inner product with the stimulus coefficients:

$$\begin{aligned} \mathbf{F}_{[t_{k^-},t_k)}(\mathbf{I}_s) &= \mathbf{F}_{[t_{k^-},t_k)}(\mathbf{c}^\top \mathbf{f}(s)) \\ &= \mathbf{c}^\top \mathbf{y}(t_{k^-}, t_k), \qquad \text{with} \\ \mathbf{y}(t_{k^-}, t_k)_i &= \mathbf{F}_{[t_{k^-},t_k)}(\mathbf{f}_i(s)) \end{aligned} \tag{4.7}$$

Ignoring the likelihood term of the first spike time t_0, we can write down the approximate log-likelihood (equation (4.3)) as follows:

$$\begin{aligned} \log p(D = \{t_1, \ldots, t_n\} | \mathbf{I}_t) = \sum_k (\alpha - 1) \log \mathbf{c}^\top \mathbf{y}(t_k, t_{k^-}) - \frac{\mathbf{c}^\top \mathbf{y}(t_k, t_{k^-})}{\beta} \\ + \log \left(\frac{\mathrm{d}\mathbf{c}^\top \mathbf{y}(t_k, t_{k^-})}{\mathrm{d}t_k} \right) + \text{const}, \end{aligned} \tag{4.8}$$

where the constant does not depend on $t_k, \mathbf{I}_t$. As Paninski pointed out [Paninski *et al.* 2007], this model is a Generalized Linear Model (GLM). The resulting encoding process is illustrated in figure 4.1.

4.3 Decoding

In the previous section, we have seen that the encoding process can be described by a conditional distribution $p(r|s)$, the probability of observing a neural response r, given that a stimulus s was presented. For the task of decoding, an important conceptual distinction can be made between *point estimation* and *probabilistic inference*. The latter consists of inferring the full posterior distribution $p(s|r)$: the probability of stimulus s, given that we observed a specific neural response r. *Point estimation* in contrast requires to make a decision for one particular stimulus as a best guess. Typical point estimates are the posterior mean $\mathbb{E}[s|r]$ or the stimulus s^* for which the posterior distribution takes its maximum (maximum a posteriori, MAP). These choices are optimal for different loss functions. A loss function specifies the 'cost' of guessing stimulus $\hat{s}$ if the true stimulus was s. The posterior mean is optimal for the squared error loss $||s - \hat{s}||^2$, whereas the MAP is optimal under the 0/1 loss. Although the 0/1 loss, which has a constant loss for arbitrarily small errors, is an arguably unnatural choice for continuous stimuli, MAP decoding is still popular and often performs well also with respect to other loss functions. Further,

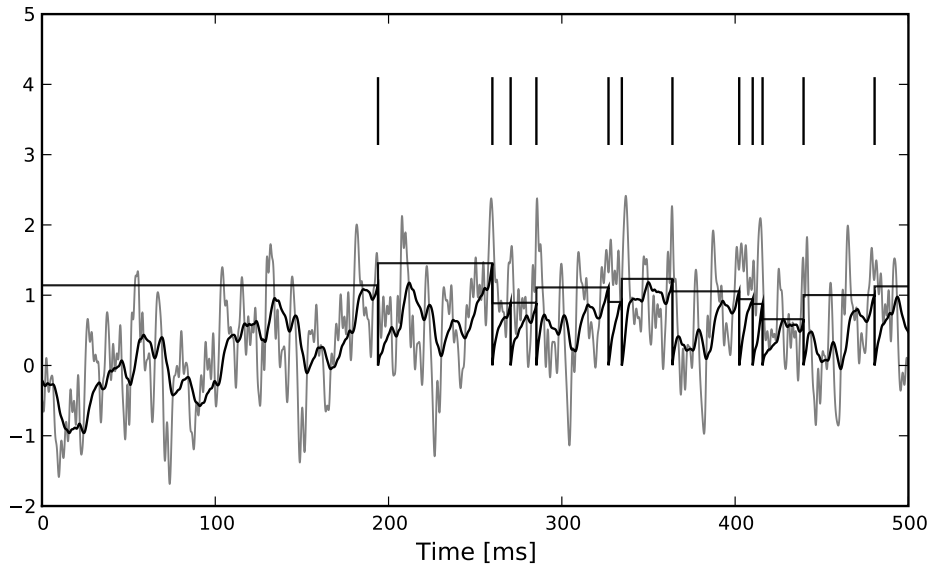

Figure 4.1: Illustration of the encoding process. We simulated a leaky ($\tau = 10$) integrate-and-fire neuron with threshold noise (mean 1.0, variance 0.05). The input is a pink noise process consisting of 80 basis functions, 40 sine and 40 cosine, frequencies equally spaced between 1 and 500 Hz. The stimulus is plotted in shaded gray, the membrane potential in black. The threshold is drawn randomly according to a gamma distribution every time a spike (vertical lines) is fired.

the posterior mean together with the posterior variance can also be regarded as a Gaussian approximation to the full posterior distribution.

In the following we will start from the noiseless case, re-deriving the pseudo-inverse decoding scheme that has been presented before by [Seydnejad & Kitney 2001]. We show that when introducing noise, the pseudo-inverse can still be seen as an approximate decoding rule, but suffers from an asymptotic bias. In order to cope with this problem, we derive a bias-reduced version as well, which can be applied in an iterative 'spike-by-spike' fashion.

4.3.1 Decoding in the noiseless case

In the noiseless case, the problem of inverting the mapping from stimulus to spike-times can be interpreted as a linear mapping [Seydnejad & Kitney 2001, Pillow & Simoncelli 2002, Arcas & Fairhall 2003]. Roughly speaking, each interspike interval defines one linear constraint on the set of possible stimuli that could have evoked the observed spike response. The evolution of the membrane potential during an interspike interval is obtained via equation (4.2). As the spike times correspond to threshold crossings of the membrane, we know that the membrane potential hits the threshold θ at time t_k:

$$\theta = \frac{1}{\tau} \int_{t_{k^-}}^{t_k} \exp\left(\frac{1}{\tau}(s - t_k)\right) \mathbf{I}_s \, \mathrm{d}s \; = \; \mathbf{F}_{[t_{k^-}, t_k)}(\mathbf{I}_s) \tag{4.9}$$

If we represent the stimulus in terms of a linear superposition of basis functions (section 4.2.2), we can address the decoding problem within the framework of finding a linear inverse mapping. Decoding of the stimulus signal $\mathbf{I}(t)$ is equivalent to inferring the coefficients $\mathbf{c}_i$ from the observed spike trains. Every interspike interval imposes a linear constraint on the coefficients $\mathbf{c}_i$.

$$\theta = \mathbf{c}^\top \mathbf{y}(t_{k^-}, t_k), \tag{4.10}$$

where the components of $\mathbf{y}$ are defined as in equation (4.7). Note that equation (4.10) is a necessary condition for the coefficients. The unknown coefficients $\mathbf{c}$ can be uniquely determined if the number of linearly independent constraints is equal to or larger than the number of unknown coefficients (see also figure 4.2). We can summarize the constraints compactly in a linear equation:

$$L\mathbf{c} = \boldsymbol{\theta}, \text{ where } \quad L := \begin{pmatrix} \mathbf{y}(t_0, t_1)^\top \\ \vdots \\ \mathbf{y}(t_{n^-}, t_n)^\top \end{pmatrix}, \quad \boldsymbol{\theta} := \begin{pmatrix} \theta \\ \vdots \\ \theta \end{pmatrix}. \tag{4.11}$$

In general, a solution to this equation can be found by using the Moore-Penrose pseudo-inverse [Penrose 1955]:

$$\mathbf{c} = L^- \boldsymbol{\theta} \tag{4.12}$$

The pseudo-inverse is well defined even if the matrix L is not square or is rank-deficient. If the number of interspike intervals exceeds the number of coefficients, the pseudo-inverse is given by

$$L^- = \left(L^\top L\right)^{-1} L^\top . \tag{4.13}$$

4.3.2 Decoding in the presence of noise

4.3.2.1 One dimensional stimulus: exact inference

We start with a simple case in which exact inference is possible: the stimulus consists of a constant (one dimensional) input c, i.e. $f_i \equiv 1$. In this situation, we can write down the likelihood exactly. For the observations we have:

$$\begin{aligned} y_k := \mathbf{y}(t_{k^-}, t_k) &= \int_{t_{k^-}}^{t_k} \exp\left(\frac{1}{\tau}(s - t_k)\right) \frac{1}{\tau} f_i(s) \mathrm{d}s \\ &= \frac{1}{\tau} \int_{t_{k^-}}^{t_k} \exp\left(\frac{1}{\tau}(s - t_k)\right) \mathrm{d}s \\ &= \frac{1}{\tau}\tau \left(1 - \exp\left(-\frac{1}{\tau}(t_k - t_{k^-})\right)\right) \\ \theta = y_k c \quad &\Rightarrow \quad \frac{\theta}{c} = y_k \end{aligned} \tag{4.14}$$

In this case, we do not have to account for the sub-threshold condition as the evolution of the membrane-potential since the last spike is a monotonic function and therefore there is only one possibility to be at the threshold for a given stimulus at a specific time. In particular, if the threshold is Gamma distributed (as assumed

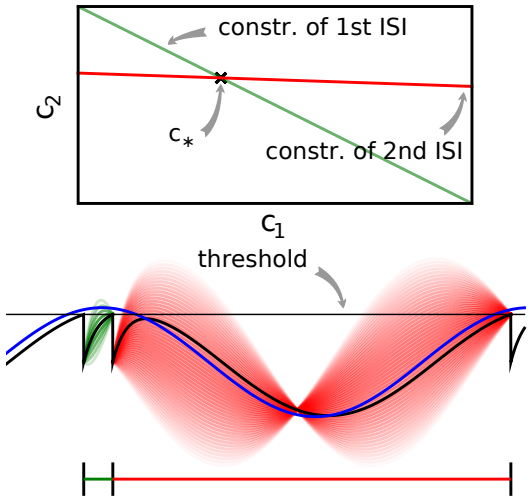

Figure 4.2: Example of noiseless decoding for a two dimensional stimulus and its limitations. The inset illustrates the linear constraints that the first and the second interspike interval pose on the two coefficients c_1 and c_2. The driving stimulus is plotted in blue.Vertical bars at the bottom indicate the three observed spike times corresponding to threshold crossings of the membrane potential (solid black). Possible membrane potential trajectories, which obey the linear constraints are plotted in shaded green and red respectively, darker ones have smaller norm. As can be seen the linear constraints only reflect that the membrane potential has to be at zero at the beginning of an interspike interval and at the threshold at the end of it. They do not reflect that the membrane potential has to stay below threshold between spike times. Parameters are: $\tau = 1$ ms, frequency for sine and cosine basis functions: 32 Hz.

in section 4.2.1), we see that $y_k|c$ is also Gamma distributed with parameters $\alpha, \frac{\beta}{c}$. For now we choose c to be Gamma distributed as well (say with parameters α_0, β_0). This choice deviates from the choice in section 4.2.2, but for this choice, we can write down the posterior exactly:

$$\begin{aligned} p(c|y_1, \ldots, y_n) &\propto \gamma(c|\alpha_0, \beta_0) \prod_k \gamma\left(y_k \middle| \alpha, \frac{\beta}{c}\right) \\ &\propto c^{\alpha_0 - 1} \exp\left(-\frac{c}{\beta_0}\right) \prod_k \left(\frac{c}{\beta}\right)^{\alpha} \exp\left(-y_k \left(\frac{c}{\beta}\right)\right) \\ &\propto c^{\alpha_0 + n\alpha - 1} \exp\left(-c\left(\sum_k \frac{y_k}{\beta} + \frac{1}{\beta_0}\right)\right) \\ &= \gamma\left(c \middle| n\alpha + \alpha_0, \left(\sum_k \frac{y_k}{\beta} + \frac{1}{\beta}\right)^{-1}\right) \end{aligned}$$

Having the posterior in closed form we can calculate the posterior mean as well as the point of maximal posterior probability exactly. Thus, we have in the special case of a constant one-dimensional input a reference for later use (see also figure 4.3).

4.3.2.2 Gaussian factor approximation

The pseudo-inverse solution of section 4.3.1 has also a probabilistic interpretation in linear Gaussian models (see also [Bishop *et al.* 2006]): In this setting, it can be interpreted as the posterior mean estimate for data with a Gaussian distribution. In particular, if (for the moment) we assume that the linear functionals $\mathbf{y}(t_{k^-}, t_k)$ are observed and that $\mathbf{c}^\top \mathbf{y}(t_{k^-}, t_k)$ is Gaussian distributed around the mean of the threshold θ with a constant variance σ_θ^2, the posterior mean of the coefficients $\mathbf{c}$ would be the same as the pseudo-inverse described above. However, this setting is not directly applicable to the context of decoding a stimulus from spike times of LIFs: In a linear Gaussian model, the observed functionals $\mathbf{y}(t_{k^-}, t_k)$ would not be allowed to depend on either c or θ, but they do here. This is most easily explained for a one-dimensional stimulus: We have that $\theta = \mathbf{cy}$, and therefore $\mathbf{y} = \theta/\mathbf{c}$. This can be highly non-Gaussian even if the distribution of θ and $\mathbf{c}$ are Gaussian[1]. We now derive a probabilistic decoding rule which is analogous to the pseudo-inverse used in the noiseless case. Each observation defines a linear constraint:

[1]The coefficient vector $\mathbf{c}$ represents the stimulus of interest and can therefore certainly not be constant.

$$\theta = \mathbf{c}^\top \mathbf{y}(t_{k^-}, t_k)$$

We can approximate the distribution of the threshold by a Gaussian term. Each linear constraint defines one factor of the likelihood. That is, p_θ in equation (4.3) is replaced with a Gaussian term of the form

$$p_\theta\left(\mathbf{y}(t_{k^-}, t_k) | t_{k^-}, \mathbf{I}_{(t_{k^-}, t_k)}\right) \approx \frac{1}{Z} \exp\left(-\frac{1}{2} \frac{\left(\mu_\theta - \mathbf{c}^\top \mathbf{y}(t_{k^-}, t_k)\right)^2}{\sigma_\theta^2}\right), \tag{4.15}$$

where $\sigma_\theta^2 = \alpha\beta^2$ is the variance of the threshold distribution. Additionally, we have replaced θ by its mean μ_θ, because we are not observing θ but t_k. Each of these factors peaks at $\mu_\theta = \mathbf{c}^\top \mathbf{y}(t_{k^-}, t_k)$, therefore reflecting the linear constraint. Replacing every term in the likelihood by its corresponding Gaussian approximation and including one Gaussian factor for the prior $p(\mathbf{c}) \sim \mathcal{N}(\boldsymbol{\mu}_c, \Sigma_c)$, the posterior is approximated by a Gaussian with the following moments:

$$\boldsymbol{\mu}_p := \left(\boldsymbol{\Sigma}_c^{-1} + \sigma_\theta^{-2} \sum_k \mathbf{y}_k \mathbf{y}_k^\top\right)^{-1} \left(\boldsymbol{\Sigma}_c^{-1} \boldsymbol{\mu}_c + \frac{\mu_\theta}{\sigma_\theta^2} \sum_k \mathbf{y}_k\right) \tag{4.16}$$

$$\boldsymbol{\Sigma}_p = \left(\boldsymbol{\Sigma}_c^{-1} + \sigma_\theta^{-2} \sum_k \mathbf{y}_k \mathbf{y}_k^\top\right)^{-1} \tag{4.17}$$

In (4.16) and (4.17), we have abbreviated $\mathbf{y}(t_{k^-}, t_k) = \mathbf{y}_k$. In addition to the pseudo-inverse (equation (4.12)), this approximation takes the prior distribution over stimuli into account, specified by the mean $\boldsymbol{\mu}_c$ and covariance $\boldsymbol{\Sigma}_c$ of the coefficients $\mathbf{c}$. This can be seen by setting $\boldsymbol{\Sigma}_c^{-1} = 0$, i. e. by using an uninformative prior. Then the mean of this approximation μ_p is exactly the pseudo-inverse of equation (4.12). Our approach of replacing likelihood factors by Gaussians is similar to the extended Kalman filter, where the dynamics is linearized and therefore results in a Gaussian update for the hidden states. However, it is known that this approximation can be biased [Julier & Uhlmann 1997, Minka 2001]. Similarly, in our case, the mean of this approximation also does not converge to the true coefficient values for increasing number of observed spikes, as shown in figure 4.3. Fortunately, under some simplifying assumptions, this bias can be calculated and therefore can be significantly reduced as will be shown in the following section.

4.3.2.3 Bias reduction of the Gaussian factor approximation

In this section we calculate the asymptotic length bias for the approximate posterior mean of equation (4.16), assuming a correct orientation of the coefficient vector. By fixing a stimulus, i. e. $\mathbf{c}$, we define the average over all resulting interspike intervals $\mathbb{E}[\mathbf{y}_k] := \boldsymbol{\mu}_y$ and $\mathrm{Cov}[\mathbf{y}_k] := \boldsymbol{\Sigma}_y$. We then find asymptotically for $n \gg 1$ and for a fixed $\mathbf{c}$:

$$\boldsymbol{\Sigma}_c^{-1} + \sigma_\theta^{-2} \sum_k \mathbf{y}_k \mathbf{y}_k^\top \longrightarrow \sigma_\theta^{-2} n \left(\boldsymbol{\Sigma}_\mathbf{y} + \boldsymbol{\mu}_\mathbf{y} \boldsymbol{\mu}_\mathbf{y}^\top \right) \tag{4.18}$$

$$\sum_k \mathbf{y}_k \longrightarrow n \boldsymbol{\mu}_\mathbf{y} \tag{4.19}$$

Note that we do not know the distribution of the $\mathbf{y}_k$ and that this distribution depends on the distribution of the threshold as well as the choice of basis functions. However, the proportion of $\mathbf{y}$ in the direction of $\mathbf{c}$ is on average of magnitude μ_θ and the variance along $\mathbf{c}$ on the other hand is σ_θ^2. Orthogonal to $\mathbf{c}$, we assume, that $\mathbf{y}$ has zero mean and finite variance. This assumption is justified in the one-dimensional case, because there is simply no orthogonal direction. Empirically, it turns out to reduce the amount of bias substantially; see figure 4.6.

Therefore, we can rewrite

$$\begin{aligned}
&\boldsymbol{\mu}_p = \left(\boldsymbol{\Sigma}_\mathbf{y} + \boldsymbol{\mu}_\mathbf{y} \boldsymbol{\mu}_\mathbf{y}^\top \right)^{-1} \left(\mu_\theta \boldsymbol{\mu}_\mathbf{y} \right) \qquad \text{where} \\
&\boldsymbol{\mu}_\mathbf{y} = \frac{\mu_\theta}{\|\mathbf{c}\|^2} \mathbf{c} \qquad \boldsymbol{\Sigma}_\mathbf{y} = \mathbf{U} \mathbf{D} \mathbf{U}^\top \qquad \text{with} \\
&\mathbf{U} = \begin{pmatrix} \frac{\mathbf{c}}{\|\mathbf{c}\|} & \mathbf{c}^\perp \end{pmatrix} \qquad \mathbf{D} = \mathrm{diag} \left(\frac{\sigma_\theta^2}{\|\mathbf{c}\|^2}, \sigma_{c_2}^2, \ldots, \sigma_{c_n}^2 \right)
\end{aligned}$$

Here, $\mathbf{c}^\perp$ denotes the basis for the space orthogonal to $\mathbf{c}$ and $\sigma_{c_1}^2, \ldots, \sigma_{c_n}^2$ are the variances in the direction of the basis vectors of $\mathbf{c}^\perp$ which are not important for the calculation of the bias. We can now compute the asymptotic posterior mean:

$$\boldsymbol{\mu}_p = \left(\mathbf{U}\mathbf{D}\mathbf{U}^\top + \frac{\mu_\theta^2}{\|\mathbf{c}\|^2}\mathbf{c}\mathbf{c}^\top\right)^{-1}\left(\frac{\mu_\theta^2}{\|\mathbf{c}\|^2}\mathbf{c}\right) \tag{4.20}$$

$$= \left(\mathbf{U}\mathbf{D}^{-1}\mathbf{U}^\top - \mathbf{U}\mathbf{D}^{-1}\mathbf{U}^\top\mathbf{c}\frac{1}{\|\mathbf{c}\|^4}\left(\mu_\theta^{-2} + \frac{1}{\sigma_\theta^2}\right)^{-1}\mathbf{c}^\top\mathbf{U}\mathbf{D}^{-1}\mathbf{U}^\top\right)\mathbf{c}\frac{\mu_\theta^2}{\|\mathbf{c}\|^2} \tag{4.21}$$

$$= \mathbf{c}\frac{\mu_\theta^2}{\mu_\theta^2 + \sigma_\theta^2} \tag{4.22}$$

We obtain (4.21) from (4.20) by using the Woodbury matrix identity. By definition of $\mathbf{U}$, all directions orthogonal to $\mathbf{c}$ cancel out and equation (4.22) follows. Equation (4.22) shows that (asymptotically) the norm of the posterior mean approximation is biased. The direction, however, is correct. Therefore, the Moore-Penrose pseudo-inverse is unbiased only in the noiseless case when $\sigma_\theta^2 = 0$. In the noisy case, however, we can divide the mean by its asymptotic bias in order to obtain an unbiased estimator for the coefficients. To improve the estimator also in the regime of few observations, we divide only the likelihood part $\frac{\mu_\theta}{\sigma_\theta^2}\sum \mathbf{y}_k$ by the asymptotic bias. Therefore we have for the bias-reduced posterior mean:

$$\hat{\mathbf{c}} = \boldsymbol{\mu}_p^{\mathrm{BC}} = \left(\boldsymbol{\Sigma}_c^{-1} + \sigma_\theta^{-2}\sum_k \mathbf{y}_k\mathbf{y}_k^\top\right)^{-1}\left(\boldsymbol{\Sigma}_c^{-1}\boldsymbol{\mu}_c + \frac{\mu_\theta^2 + \sigma_\theta^2}{\mu_\theta^2}\frac{\mu_\theta}{\sigma_\theta^2}\sum_k \mathbf{y}_k\right). \tag{4.23}$$

This bias-reduced version of the Moore-Penrose inverse is also plotted in figure 4.3, which gives an improved estimate also for a small number of observations. The presented bias-reduced Gaussian approximation can also be rewritten into an online algorithm. The update equations to incorporate one additional observation $\mathbf{y}_k$ in the current posterior are given by

$$\boldsymbol{\mu}_p^{k+1} = \boldsymbol{\mu}_p^k - \boldsymbol{\Sigma}_p^k\mathbf{y}_k\left(\sigma_\theta^2 + \mathbf{y}_k^\top\boldsymbol{\Sigma}_p^k\mathbf{y}_k\right)^{-1}\left(\mathbf{y}_k^\top\boldsymbol{\mu}_p^k - \mu_\theta\right) \tag{4.24}$$

$$\boldsymbol{\Sigma}_p^{k+1} = \boldsymbol{\Sigma}_p^k - \boldsymbol{\Sigma}_p^k\mathbf{y}_k\left(\sigma_\theta^2 + \mathbf{y}_k^\top\boldsymbol{\Sigma}_p^k\mathbf{y}_k\right)^{-1}\mathbf{y}_k^\top\boldsymbol{\Sigma}_p^k. \tag{4.25}$$

Together with equation 4.23 we thus obtain a bias-reduced on-line estimator which allows one to recursively improve the stimulus reconstruction on a spike-by-spike basis:

$$\boldsymbol{\mu}_p^{k+1} = \boldsymbol{\mu}_p^k - \boldsymbol{\Sigma}_p^k \mathbf{y}_k \left(\sigma_\theta^2 + \mathbf{y}_k^\top \boldsymbol{\Sigma}_p^k \mathbf{y}_k\right)^{-1} \left(\mathbf{y}_k^\top \frac{\mu_\theta^2}{\mu_\theta^2 + \sigma_\theta^2} \boldsymbol{\mu}_p^k - \mu_\theta\right) \frac{\mu_\theta^2 + \sigma_\theta^2}{\mu_\theta^2} \tag{4.26}$$

We can now compare how well the different approximations perform compared to the exact solution in the one-dimensional case (see section 4.3.2.1). In figure 4.3 the mean squared error is shown as function of the number of observed interspike intervals. Plotted are the error of the MAP estimator (magenta), the exact minimum mean squared error (blue), the Gaussian-Factor approximation, which is the equivalent to the Moore-Penrose pseudo-inverse (red) and the bias reduced Gaussian-Factor approximation (green). Importantly, the solution obtained by the Moore-Penrose pseudo-inverse does not converge to the true solution, but has a strong bias. This bias can lead to a solution which is actually worse than the prior solution. Unfortunately, we do not have access to the exact posterior in general, especially in higher dimensions. Therefore, we need approximation schemes which are generally applicable in the general case, but which perform better than the Moore-Penrose pseudo-inverse.

4.3.3 Two-dimensional case

In section 4.3.2.1, we investigated the accuracy of the different reconstruction schemes in the one-dimensional case. If the stimulus is two- or higher-dimensional, the observation of a single spike does not give us full rank information about the stimulus. In the case of a two-dimensional stimulus, three types of scenarios can occur after one interspike interval has been observed:

1. The observation of an interspike interval only leads to one important constraint on the coefficients of the basis functions, namely that the membrane potential has to be at the threshold at the time of a spike. For example, if the observed interspike interval is relatively small, solutions which cross the threshold twice or hit the threshold from above, are very unlikely under the prior distribution. Therefore, to stay below threshold, one can neglect constraints other than being at the threshold at the time of the observed spike, see also figure 4.2. In this situation, all approximations should be almost equally good as they all account for this type of constraint.

2. If the interspike interval is longer, we might get another important constraint for the posterior, namely by requiring that the threshold is hit from below, not from above. This possibility is ruled out by the Jacobian term of the

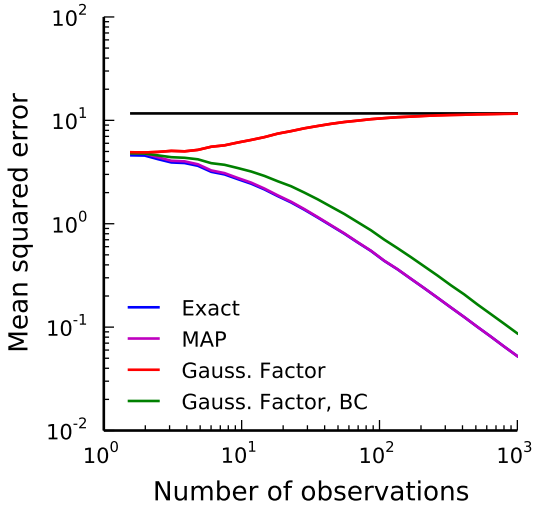

Figure 4.3: Comparison of the mean squared error (MSE) for different reconstruction methods in the case of a one dimensional stimulus. The best possible estimate is the true posterior mean (exact, blue). The error of the maximum a posteriori (MAP) estimator (magenta) is nearly the same as the error of the exact posterior mean and therefore cannot be distinguished from the exact one. The red line shows the error of the Moore-Penrose pseudo-inverse and the horizontal line indicates its asymptotic bias. The Moore-Penrose pseudo-inverse is called Gaussian Factor approximation (see section 4.3.2.2). The bias corrected (BC) version of the Gaussian approximation (green) is explained later and here included for completeness (see section 4.3.2.3). Parameters were: $\alpha_{\text{prior}} = 20, \beta_{\text{prior}} = 0.5, \alpha_\theta = 2, \beta_\theta = 0.5$

pseudo-likelihood (4.32). Therefore the MAP estimate should be closer to the true posterior mean than the Gaussian or pseudo-inverse approximation, which does not satisfy this constraint. Here, crossing the threshold twice before hitting it again from below is still very unlikely according to the prior and therefore we do not get an effective restriction for the posterior by ruling out all these solutions which cross the threshold twice.

3. If the interspike interval is sufficiently long, both types of violations of crossing the threshold between spike times are probable according to the prior. Some possible stimuli might exist for which the membrane potential would cross the threshold twice before reaching the threshold again at the time of the observed spike. These stimuli are neither ruled out by the pseudo-likelihood nor by the Gaussian approximation. Therefore, both approximations can result in quite poor estimates of the true posterior mean.

To illustrate the three scenarios, we simulated a single neuron with a stimulus consisting of two basis functions, one sine and one cosine function. We obtained an approximation to the true posterior after single observations by rejection sampling. This true posterior reflects all of the constraints mentioned above. As can be seen in figure 4.4, indeed three types of situations can be observed.

4.4 Alternative methods

In the following, we will discuss the relationship between our decoding rule and previously proposed decoding algorithms. In particular, we compare our decoders with an optimal linear decoder, as well as with a Maximum-a-Posteriori decoder (MAP) based on the approximate likelihood.

4.4.1 Relationship to the linear decoder

Bialek et al. popularized a linear decoder for reconstructing the stimulus from a spike train [Bialek *et al.* 1991, Rieke *et al.* 1997]. Here the spike train $\sum_i \delta(t - t_i)$ is convolved with an acausal linear filter K in order to obtain an estimate of the stimulus:

$$\hat{s}(t) = \sum_i K(t - t_i) = K \star \sum_i \delta(t - t_i) \tag{4.27}$$

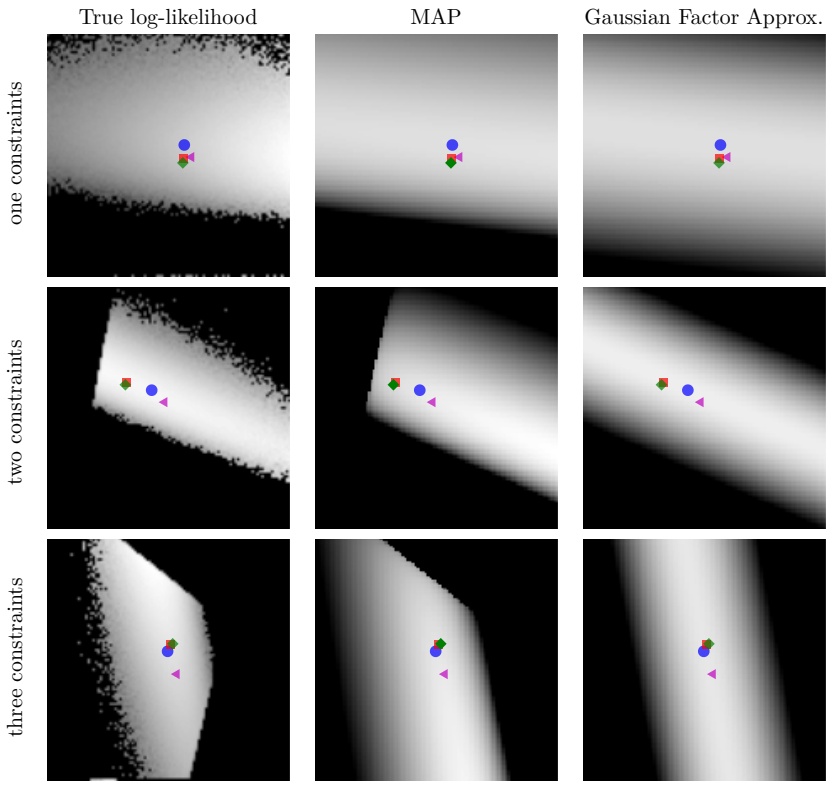

Figure 4.4: Log-likelihood approximations in two dimensions for three different cases of observations and different approximations to the posterior. The first column is the true log-likelihood, the second is the approximate log-likelihood obtained by equation (4.32) and the third column is the Gaussian Factor approximation. The true log-likelihood is not available in higher dimensions and is plotted here for comparison and as a reference. It is obtained via rejection sampling. Point estimates are: true posterior mean (●), MAP (◀), Gaussian Factor mean (■) and the bias reduced version (♦). For each point estimate a Gaussian prior with unit isotropic covariance was chosen. Each subplot shows the log-likelihood (or its approximation) after one interspike interval is observed. The x and y axes indicate the two dimensions of the stimulus coefficients. Each row corresponds to a different scenario with different numbers of effective constraints for the posterior. If only one constraint is active (first row) the true posterior does not differ much from the other approximations, and therefore the point estimates perform all almost equally well. If two constraints are active (the threshold has to be reached from below and the membrane potential has to be at the threshold at the time of a spike) the MAP performs better than the Gaussian Factor approximation. If three constraints are active, the MAP reflects two of the three constraints and therefore is slightly shifted. As one observation is far away from the asymptotic regime, the Gaussian Factor approximation and its bias reduced version do not differ much.

The filter can be calculated by [Rieke *et al.* 1997]:

$$\mathcal{F}(K)(\omega) = \frac{\mathbb{E}[\mathcal{F}(s)(\omega)\sum_k \exp(i\omega t_k)]}{\mathbb{E}\left[\left|\sum_k \exp(i\omega t_k)\right|^2\right]}, \tag{4.28}$$

where $\mathcal{F}$ is the Fourier transform. The average is taken over the joint distribution of stimuli and spike times, which can be done via sampling. Additionally, the stimuli we used are composed by a superposition of sine and cosine functions with discrete frequencies, which we write here as complex functions $f_l(t) = \exp(i\omega_l t)$. Hence, the linear filter has also only non-vanishing power in those frequencies which are present in the stimulus.

In the noiseless case, the Pseudo-Inverse decoder can be interpreted as a linear filter, but one that depends on the particular spike train observed, as we will show in the following. To this end, we replace the stimulus ensemble used to calculate the linear filter with a single stimulus consisting of the stimulus reconstructed by the Pseudo-Inverse. That is, we replace $\mathcal{F}(s)(\omega_l)$ by $\sum_j \mathbf{L}^-_{i,j}\theta$; see equation (4.12). If we further assume that there is no neuronal noise, we can neglect the expectation in the definition of the linear filter (equation (4.28)), and define a linear filter K_p corresponding to the Pseudo-Inverse:

$$\begin{aligned} \mathbf{c}_j^{K_p} &:= \frac{\sum_k L^-_{j,k}\theta}{\sum_k \exp(-i\omega_j t_k)} \\ &= \frac{\sum_k L^-_{j,k}\theta(\sum_k \exp(i\omega_j t_k))}{|\sum_k \exp(-i\omega_j t_k)|^2} \end{aligned} \tag{4.29}$$

Although this equivalence is only valid in the noiseless case, we can use equation (4.29) to illustrate the decoding performed by the Pseudo-Inverse. The linear filters we obtain for this decoder is different for different spike trains, reflecting the increased flexibility of the Pseudo-Inverse compared to the optimal linear predictor. The different reconstructions and associated filters are illustrated in figure 4.5.

4.4.2 Maximum a posteriori and Laplace approximation

By inspecting the approximate likelihood (see equation (4.8)) we see that the model is a generalized linear model. In this sense it is very similar to the soft-threshold noise model [Paninski *et al.* 2007, Jolivet *et al.* 2006]. However, the threshold noise there is Poisson-like, whereas here it is Gamma distributed. Further, the soft-threshold likelihood does not account for the fact that the threshold has to be reached from below. By ensuring that the Jacobian of the change of variables in

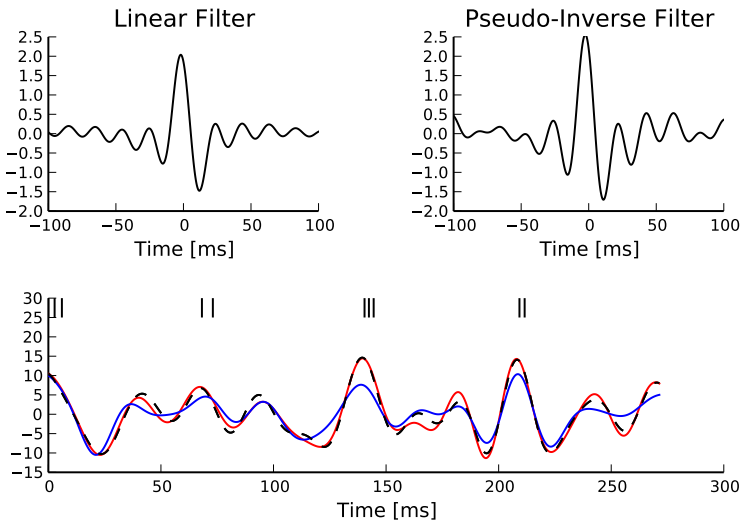

Figure 4.5: Comparison of the linear decoder and the Gaussian factor approximation. **Upper left**: Linear filter obtained via equation (4.28). **Upper right**: Average linear filter for the Pseudo-Inverse or Gaussian factor approximation, see equation (4.29). **Bottom**: Example of a decoded stimulus for a given spike train by two decoding schemes. The true stimulus is plotted in dashed black, the Gaussian factor reconstruction in red and the linear decoder reconstruction is plotted in blue. Shown are a window of the first 10 out of 100 spikes. The stimulus consisted of 20 sine and 20 cosine functions with frequencies between 10 and 50 Hz. Spikes are generated with a leaky integrator with time constant $\tau = 25$ms. The noise is relatively low: $\sigma_\theta^2 = 0.01, \mu_\theta = 1$. The squared errors for the trial here are: 3.27 for the linear decoder and 2.11 for the pseudo-inverse.

equation (4.3) is positive, however, we can take this constraint into account. One approach for getting a possibly better point estimate is to find the maximum of the approximate posterior density (MAP). To compute this posterior density, we have to multiply equation (4.3) by the prior density (which is Gaussian in our case). In this model, the MAP cannot be determined in closed form, but we may apply gradient ascent in order to find it numerically. If both likelihood and prior are log-concave, which is true for the approximate likelihood and the Gaussian prior used here, the posterior is unimodal [Paninski *et al.* 2004]. Hence, finding the MAP point is a convex problem. The gradient and the Hessian of the log posterior are straightforward to compute. For the sake of clarity, we only write down the gradient and the Hessian for one spike time t_k in the sum of equation (4.8):

$$\begin{aligned}\nabla_{\mathbf{c}}(k) &= \frac{(\alpha-1)}{\mathbf{c}^\top \mathbf{y}(t_k, t_{k-1})}\nabla_{\mathbf{c}}\mathbf{c}^\top \mathbf{y}(t_k, t_{k-1}) - \frac{1}{\beta}\nabla_{\mathbf{c}}\mathbf{c}^\top \mathbf{y}(t_k, t_{k-1}) \\ &+ \left(\frac{\mathrm{d}\mathbf{c}^\top \mathbf{y}(t_k, t_{k-1})}{\mathrm{d}t_k}\right)^{-1} \frac{\mathrm{d}\nabla_{\mathbf{c}}\mathbf{c}^\top \mathbf{y}(t_k, t_{k-1})}{\mathrm{d}t_k} \\ &= \left(\frac{\alpha-1}{\mathbf{c}^\top \mathbf{y}(t_k, t_{k-1})} - \frac{1}{\beta}\right)\mathbf{y}(t_k, t_{k-1}) \\ &+ \left(\mathbf{c}^\top(\mathbf{f}(t_k) - \mathbf{y}(t_k, t_{k-1}))\right)^{-1}(\mathbf{f}(t_k) - \mathbf{y}(t_k, t_{k-1}))\end{aligned} \tag{4.30}$$

Here, $\mathbf{f}(t_k)$ is the vector consisting of all basis functions evaluated at the spike time t_k. The Hessian is given by:

$$\begin{aligned}\nabla^2_{\mathbf{c}}(k) = &-\mathbf{y}(t_k, t_{k-1})\left(\frac{\alpha-1}{(\mathbf{c}^\top \mathbf{y}(t_k, t_{k-1}))^2}\right)\mathbf{y}(t_k, t_{k-1})^\top \\ &- (\mathbf{f}(t_k) - \mathbf{y}(t_k, t_{k-1}))\left(\mathbf{c}^\top(\mathbf{f}(t_k) - \mathbf{y}(t_k, t_{k-1}))\right)^{-2}(\mathbf{f}(t_k) - \mathbf{y}(t_k, t_{k-1}))^\top\end{aligned} \tag{4.31}$$

Applying a gradient ascent scheme yields a point estimate that respects the constraint that the membrane potential crosses the threshold from below. Nevertheless, it does not take into account the sub-threshold condition between spike times: The solution we get might correspond to a membrane potential that crosses the threshold twice before it hits it again from below. Therefore this point estimate suffers from the same source of bias as the Gaussian factor approximation.

This point estimate can be extended to give an approximation of the uncertainty as well by expanding the posterior to second order around the MAP point. The

posterior we are using here is the likelihood (equation (4.3)) times a prior term $p(\mathbf{c})$:

$$p(\mathbf{c}|\{t_0,\ldots,t_n\}) \approx p(\mathbf{c})p(t_0|\mathbf{I}_{(0,t_0)}) \prod_{k=1}^{n} p_\theta(\mathbf{y}(t_k,t_{k-1})|t_{k-1},\mathbf{I}_{(t_{k-1},t_k)}) \left|\frac{\mathrm{d}\mathbf{c}^\top \mathbf{y}(t_k,t_{k-1})}{\mathrm{d}t_k}\right|, \tag{4.32}$$

which itself is an approximation, see section 4.2.1. Unfortunately, computing the normalization constant for this distribution with respect to $\mathbf{c}$ is not tractable. We therefore approximate the posterior by a second-order expansion. In other words, the posterior distribution is approximated by a multivariate Gaussian, where the mean of the Gaussian is taken to be the MAP, and the covariance is found by looking at the second-order derivatives of the log-posterior at the MAP:

$$\begin{aligned}
\mathbf{c}_{\mathrm{MAP}} &:= \arg\max_c p(\mathbf{c}|D) \\
H^{-1} &:= -\nabla^2_{\mathbf{c}} \log p(\mathbf{c}|D) \\
p_{\mathrm{Laplace}} &:= \mathcal{N}(\mathbf{c}_{\mathrm{MAP}}, H) = \frac{1}{(2\pi)^{(M/2)}|H|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{c}-\mathbf{c}_{\mathrm{MAP}})^\top H^{-1}(\mathbf{c}-\mathbf{c}_{\mathrm{MAP}})\right).
\end{aligned}$$

The MAP and the Hessian are calculated by equation (4.30) and (4.31). This yields a Gaussian approximation known as the Laplace approximation [Paninski *et al.* 2007, MacKay 2003, Cunningham *et al.* 2008, Rasmussen & Williams 2006].

4.5 Simulations

In this section, we present the results of three different simulations which highlight different aspects of neural population coding of time-varying stimuli with integrate and fire neurons. As a general framework, we first specify a generative model for the stimulus signal $\mathbf{x}(t) = (x_1(t),\ldots,x_m(t))$ and then we specify a mapping $g : \mathbf{x}(t) \mapsto \mathbf{I}(t)$, which can be interpreted as the encoding strategy of the neural population. The dimension of $\mathbf{I}(t) = (I_1(t),\ldots,I_n(t))$ can be different from the number of spatial stimulus components m. Each $I_i(t)$ represents one neuron within a population of n neurons. Each spatial component $x_l(t), l = 1,\ldots,m$ is represented with a superposition of temporal basis functions $f_k(t), k = 1,\ldots,M$. In the first simulation we have $n = m = 1, M = 80$. In the second and third simulation $n \gg m = 1, M = 40$. In the last simulation we study the encoding of an amplitude and phase variable with $n = m = 2, M = 40$.

4.5.1 One neuron, one component, many temporal dimensions

In order to evaluate the accuracy of our Gaussian Factor approximation to the posterior when the stimulus has several temporal dimensions (not to be confused with spatial dimensions m), we analyzed the decoding performance as a function of increasing number of observations. To this end, we simulated a neuron with a stimulus consisting of a random superposition of 40 sine and 40 cosine functions with equally spaced frequencies between 10 and 50 Hz. In each trial the neuron was simulated until 10^4 spikes were accumulated. We calculated the mean squared error over 100 repetitions. Interspike intervals taken into account for reconstruction were randomly selected from the whole time interval of the simulation.

In figure 4.6 we see that the simple Gaussian approximation (Gaussian Factor, red, dashed) is indeed biased and the bias is larger for larger noise levels. In the limit of no noise we expect a sharp drop off for the number of spikes equal to the number of dimensions for the stimulus. This is weakened in the presence of noise. For comparison, we also plot the asymptotic error of the Gaussian Factor approximation as derived analytically in section 4.3.2.3. Additionally the mean squared errors are plotted for the linear decoder (see section 4.4.1) and the bias-reduced version of the Gaussian approximation. The mean squared error for the MAP was obtained by gradient ascent, see section 4.4.2. In order to start with a feasible solution, we initialized the optimizer with the true stimulus coefficients, turning the obtained solution in an optimistic estimate of the actual MAP.

4.5.2 Many neurons, many temporal dimensions

In this simulation, a population of $n = 30$ neurons with different receptive fields were all driven by the same stimulus, which consisted of a superposition of 20 sine and 20 cosine functions $x(t) = \sum_{k=1}^{20} c_{2k-1} \sin \omega_k t + c_{2k} \cos \omega_k t$. The frequencies $\{\omega_k\}_{k=1}^{20}$ were equally spaced between 1 and 100 Hz, and the coefficients $\{c_j\}_{j=1}^{40}$ were drawn independently from a Gaussian distribution with unit variance.

Incorporating a receptive field $r^i(t)$ for neuron i in our model can easily be done by pre-filtering the stimulus with the corresponding receptive field:

$$\mathbf{I}_t = (r^1 \star x(t), \ldots, r^n \star x(t))^\top$$

Because of the linearity of the convolution, the decoding algorithms stay the same with the exception that the basis functions $f_k(s)$ are replaced by $r^i \star f_k$. The

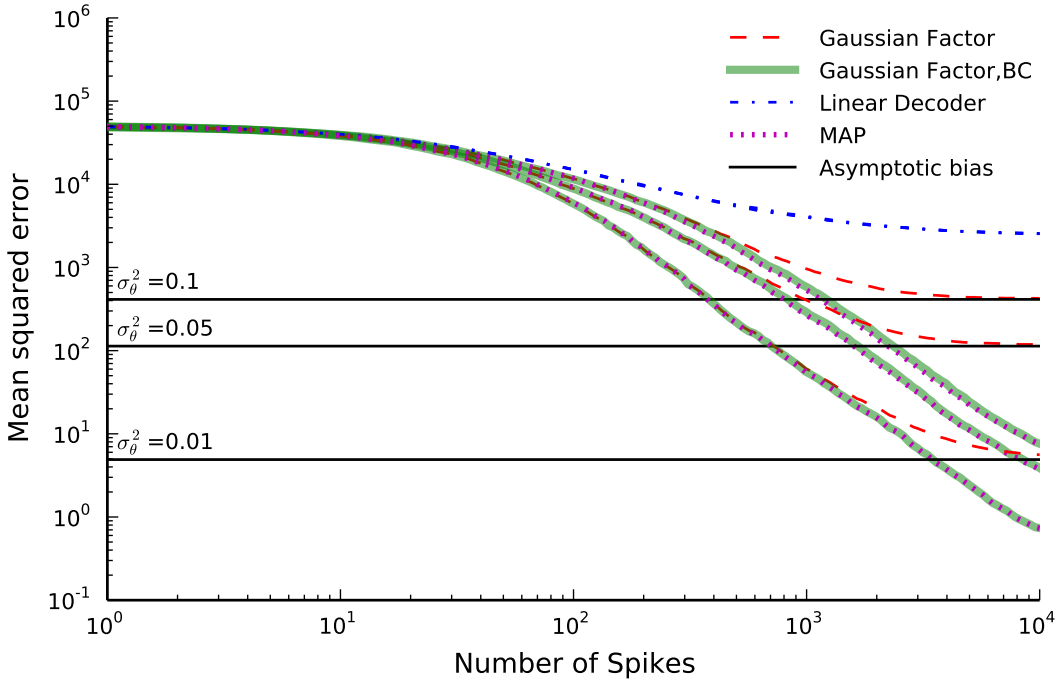

Figure 4.6: Mean squared error (MSE) as a function of the number of spikes used for the different decoding schemes. The stimulus consists of a superpostion of 40 sine and 40 cosine functions of discrete frequencies equally spaced between 10 and 50 Hz. The time constant of the neuron used for decoding is $\tau = 25$ ms. The MSE is calculated as the average over 100 repetitions for three different noise levels. Horizontal lines indicate the asymptotic bias for the different noise levels. The prior was an isotropic Gaussian with zero mean and covariance matrix $\mathbb{1} \cdot 25$.

receptive fields $r^i(t)$ of each neuron were chosen to be a gamma tone:

$$r^i(t) = at^{n-1}\cos(2\pi f_i t + \phi)\exp(-2\pi bt)$$

All parameters except the frequency f were fixed ($a = 0.01, b = 0.01\ [\frac{1}{\text{ms}}], n = 2, \phi = 0$). The frequencies of each receptive field were drawn from a uniform distribution ranging between 1 and 100 Hz. The resulting receptive fields are shown in figure 4.7(a). The stimulus and its reconstruction based on the spike times of this population are shown in figure 4.7(b). The uncertainty is smaller within periods of higher firing rates, yet to a smaller extent than in the next setting (see figure 4.9), because here the receptive fields have a larger temporal extent.

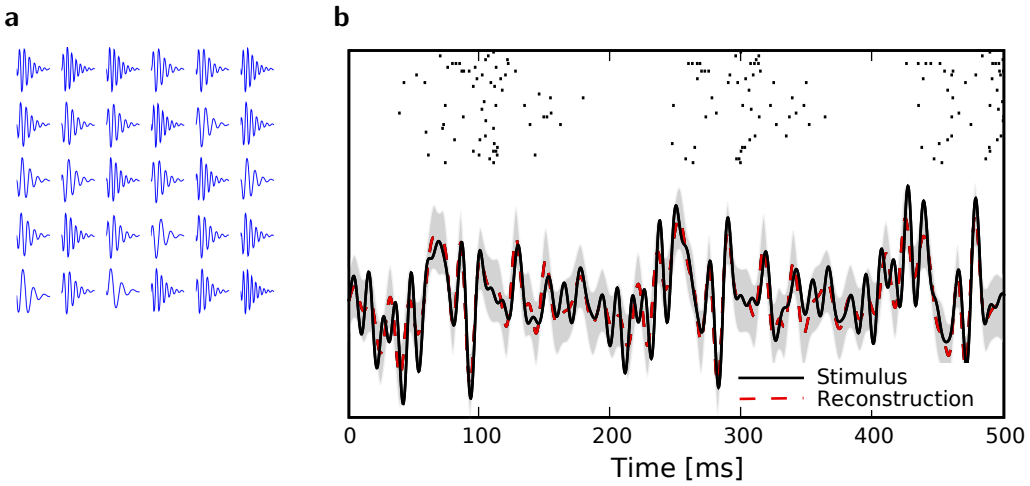

Figure 4.7: **a:** Receptive fields of the population, each is a gamma tone with a different frequency, randomly drawn from a uniform distribution between 1 and 100 Hz. **b:** A time varying stimulus consisting of a superposition of 20 sine and 20 cosine functions is decoded from spike trains of a population of 30 neurons, each having a noise level of σ_θ=0.05.

4.5.3 Heterogeneity across the population

Every new spike contributes new information about the stimulus, and leads to a reduction in reconstruction error. However, if the resulting linear constraints are correlated, the reduction can be arbitrarily small. This problem can become particularly severe for interspike intervals observed at different neurons. For example, if the parameters of different neurons (e. g. the receptive fields) are the same, spikes of different neurons tend to synchronize, even in the presence of threshold noise. This leads to similar interspike intervals, and thus to highly correlated linear constraints.

In this case, the information conveyed by different neurons can be redundant and be of limited use for decoding.

It is plausible that efficient population codes should have heterogeneity in their receptive field properties, to ensure that different properties of the stimulus are sampled by the population. In our setting, diversity in receptive field parameters would ensure that the constraints are less correlated and that the reconstruction error does not saturate with increasing numbers of neurons. As a result, we expect to get a better reconstruction if we have a larger diversity within the encoding population.

In this simulation, we extend the previous example by systematically varying the degree of similarity in the receptive field properties among the different neurons. To construct heterogeneous populations with different degrees of diversity we sampled the center frequencies of the receptive field (gamma tone) of each neuron from a uniform distribution within a frequency interval centered at 50 Hz (the center frequency of the stimulus used). The degree of diversity was then measured by the length of this interval, from 0 to 25 Hz. Figure 4.8 shows the mean squared error as a function of number of neurons as well as the diversity within the receptive fields. From this plot one can see that the rate with which the error drops with increasing number of neurons strongly depends on the degree of diversity. This result confirms the general idea of redundancy reduction as an efficient coding strategy.

4.5.4 Encoding of amplitude and phase variables

In this simulation we consider the case of decoding a two-dimensional, time-varying stimulus signal. In particular, we want to illustrate how the encoding of angular variables can be addressed in this framework, as the neural representation of edge orientations or motion directions are frequently studied in neuroscience. Therefore, we use the nonlinear polar coordinate transform to obtain an amplitude and phase variable $\mathbf{x}(t) = (a(t), \varphi(t))^\top$ as our stimulus signal. For simplicity, we consider the case where this signal is encoded by two neurons with identical temporal receptive field properties but with 90° difference in the preferred stimulus angle. Specifically, the encoding model of the two neurons is given by

$$\mathbf{I}(t) = a(t) \begin{pmatrix} \sin \varphi(t) \\ \cos \varphi(t) \end{pmatrix}.$$

As temporal basis functions we picked 20 sine and cosine basis functions with discrete equally spaced frequencies between 1 and 10 Hz. The corresponding coef-

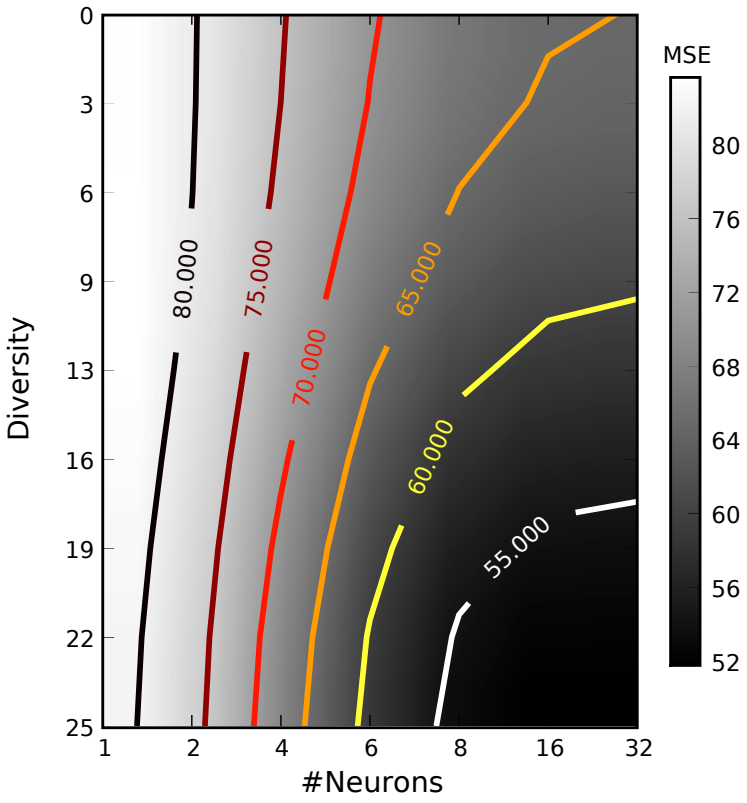

Figure 4.8: Mean squared error as a function of the number of neurons and their diversity within their receptive fields. Diversity is measured by the width of the uniform distribution from which frequencies for the gamma tone receptive fields were drawn. The average is taken over ≥ 25 repetitions. All other parameters were as in the previous section.

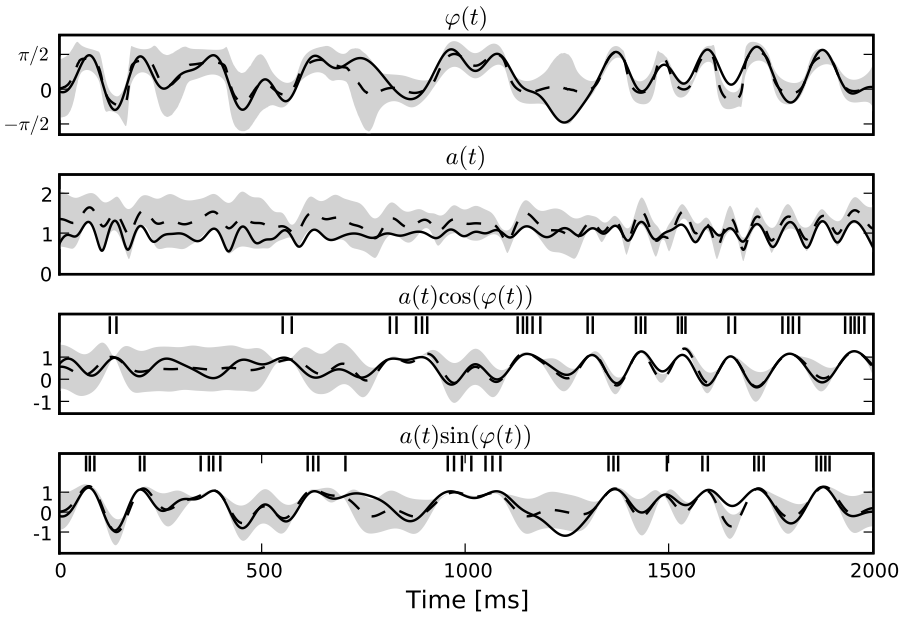

Figure 4.9: Decoding of an angular variable. Two neurons were stimulated with $a(t)\sin\varphi(t)$ and $a(t)\cos\varphi(t)$, respectively (two bottom panels). Each of those signals was represented by a superposition of 20 sine and 20 cosine functions. From the reconstructed signal, the amplitude $a(t)$ and the phase angle $\varphi(t)$ were obtained by taking the Euclidean norm and the arc-tangent, respectively. The reconstruction (dashed) of the original stimulus (solid) was obtained by using the Gaussian approximation with bias correction. Confidence intervals, indicating one standard deviation of the posterior variance, are plotted in shaded gray. The confidence intervals of $a(t)$ and $\varphi(t)$ were calculated by drawing 5000 samples from the approximate posterior.

ficients c_k were drawn independently from a Gaussian distribution with variance[2] $\sigma^2 = 0.06$. The neurons were simulated according to equation (4.1), with parameters $\tau = 10, \mu_\theta = 1, \sigma_\theta^2 = 0.01$. As can be seen from figure 4.9, the two dimensional signal (bottom two panels) can be reconstructed best in those time intervals which contains spikes (vertical black lines). The reconstruction and uncertainty (obtained via sampling) are transformed into phase and amplitude in the top two panels.

4.6 Discussion

How to read out spatio-temporal spike patterns generated by populations of neurons is fundamental to the understanding of neural network computation. Most of

[2]The small variance was chosen such that the resulting signal varies roughly between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$

the previous studies on population coding were limited to the static case where only spike counts for a preset time window are considered. For the encoding of continuously varying signals, however, it is important to understand how the accuracy of population codes is affected by the dynamics of neural spike generation.

Here, we studied dynamic population codes with noisy leaky integrate-and-fire neurons. We presented an algorithm for Bayesian decoding similar to the one presented in [Cunningham *et al.* 2008]. In addition, we derived an approximate algorithm which yields a simple spike-by-spike update rule for recursively improving the stimulus reconstruction whenever a new spike is observed.

The decoding rules can also be applied for decoding the spike trains of populations of neurons, not just single neurons. Importantly, we do not have to assume that the neurons are uncoupled, i. e. conditionally independent given the stimulus. In particular, as we assume the encoding model to be known, we would also know the parameters describing the couplings between neurons. Then, the influence of one spike of a neuron on the membrane potential of any other neuron is just a known, given input and can be subtracted. Therefore, the same decoding framework can also be used for decoding coupled neurons.

The decoding rule is nonlinear and sensitive to the relative latencies between each spike and its predecessor in the population. However, it is not optimal as it does not use the information that the membrane potential stays below threshold between spikes. To incorporate this kind of knowledge one has to integrate the coefficient distribution over the linear halfspace confined by the threshold similar to the method described in [Paninski *et al.* 2004, Paninski *et al.* 2007] but with the additional complication that, the distribution is not Gaussian. Therefore, the optimal Bayesian decoding rule would be computationally much more expensive.

The main goal of this work was to derive a simple decoding rule that facilitates the analysis of neural encoding strategies such as efficient coding, unsupervised learning, or active sampling. Bayesian approaches are particularly useful for these problems as they do not yield a point estimate only but also aim at estimating the posterior uncertainty over stimuli. Having access to this uncertainty allows one to optimize receptive field properties or other encoding parameters in order to minimize the reconstruction error or to maximize the mutual information between stimulus and neural population response. In this way it becomes possible to extend unsupervised learning models such as independent component analysis [Bell & Sejnowski 1995] or sparse coding [Olshausen & Field 1996] to the spatio-temporal domain with spiking neural representations. This seems highly desirable as comparisons between theoretically derived models and experimental measurements would thus become feasible.

Furthermore, animals do not receive the sensory input in a passive way but actively tune their sensory organs to acquire the most useful data, for example by changing gaze or by head movements. Such active sampling strategies are related to the theory of optimal design or active learning [Lewi *et al.* 2008], where the next measurement is selected in order to minimize the current uncertainty about the signal of interest. Such active sampling strategies give rise to 'saliency maps', which encode the expected information gain from any particular stimulus.

Maximizing the mutual information between stimulus and neural response is equivalent to minimizing the posterior entropy. Because of the Gaussian approximation, this can be done in our model by performing a gradient descent on the log-determinant of the posterior covariance matrix. The gradient can be calculated from equation (4.17). However, the approximated posterior covariance derived in this chapter might also be subject to a systematic deviation from the exact covariance matrix. Therefore, an important extension of the present work would be to correct for a bias in the approximate covariance estimate, too. In general the approximations considered in this chapter usually tend to over-estimate the true underlying uncertainty, as they wrongly do not cut-off regions in the parameter space.

In this work, we chose to represent the stimulus by a superposition of a finite set of basis functions as this has some practical advantages. Alternatively, it is also possible to start from a full Gaussian process as stimulus model and then derive a discretization for numerical evaluation. Analogous to the mean vector and covariance matrix of a finite-dimensional normal distribution, a Gaussian process prior over the stimulus is specified by the mean and covariance function of the process. For numerical evaluation it is necessary to choose a grid of time points yielding a finite dimensional normal distribution again.

Note that for inference, integrals on the grid points have to be evaluated numerically and therefore a fine time resolution for the s_i should be chosen. Therefore, the computational load of decoding a discretized Gaussian process is considerably higher. For practical reasons, we can restrict the inference procedure to a time window around the current spikes, provided that the covariance function falls off quickly. In the non-leaky case with no receptive fields this is the same setting as in [Cunningham *et al.* 2008].

The extension to the Gaussian process setting is conceptually important as it allows one to replace the somewhat artificial threshold noise model by membrane potential noise. The dynamics can then be described by a stochastic differential equation. Although the likelihood is much harder to calculate [Paninski *et al.* 2004, Paninski *et al.* 2007], it still has the renewal property and

therefore a similar approximation scheme might be applicable. However, it has the further complication, that the obtained likelihood is only for a given threshold and therefore the threshold has to be marginalized. We hope that more studies will be devoted to the problem of decoding time-varying stimuli from populations of spiking neurons in the future. In particular, it will be crucial to achieve a good trade-off between the basic dynamics of neural spike generation, the accuracy of posterior estimates and the computational complexity of the decoding algorithm.

5

Joint modeling of stimuli and population responses

5.1 Introduction

In chapter 3 we have modeled the encoding distribution $p(\mathbf{r}|\mathbf{s})$ of a neural response $\mathbf{r}$ given the stimulus $\mathbf{s}$ was presented. In order to reverse this relationship to obtain the decoding distribution $p(\mathbf{s}|\mathbf{r})$ we had to assume a prior distribution over stimuli $p(\mathbf{s})$. However, describing the statistics of natural stimuli is a nontrivial task. In general, both distributions could be calculated if we had access to the joint distributions of responses and stimuli, that is if we knew how likely a pair of stimuli and a population response is to be observed.

Not only modeling the response or the stimulus but the distribution of both variables results in much more degrees of possible variation. To estimate such high-dimensional distributions requires collecting massive amounts of data. Recent technical advances in systems neuroscience allow us to monitor the activity of increasingly large neural ensembles simultaneously (e.g. [Buzsaki 2004, Shlens *et al.* 2009]). To understand how such ensembles process sensory information and perform the complex computations underlying successful behavior also the use of suitable statistical models for data analysis are needed. What degree of pre-

cision should be incorportated into such a model involves a trade-off between the question of interest and mathematical tractability.

Maximum entropy modeling has been successfully applied in a number of disciplines such as physics, computer vision and natural language processing. The reasoning behind this principle is that if a probability distribution is underdetermined by the data one should choose from all distributions which are consistent with the data the one, which has maximum entropy. Recently, several groups have used binary maximum entropy models incorporating pairwise correlations to model neural activity in large populations of neurons on short time scales [Schneidman *et al.* 2006, Shlens *et al.* 2006, Tang *et al.* 2008, Yu *et al.* 2008]. These models have two important features: (1) Since they only require measuring the mean activity of individual neurons and correlations in pairs of neurons, they can be estimated from moderate amounts of data. (2) They seem to capture the essential structure of neural population activity at these timescales even in networks of up to a hundred neurons [Shlens *et al.* 2009]. Although the generality of these findings have been subject to debate [Bethge & Berens 2008, Roudi *et al.* 2009b], pairwise maximum-entropy and related models [Macke *et al.* 2009] are an important tool for the description of neural population activity [Shlens *et al.* 2008, Roudi *et al.* 2009a].

To find features to which a neuron is sensitive spike-triggered average and spike-triggered covariance are commonly used techniques [Schwartz *et al.* 2002, Pillow & Simoncelli 2006]. They correspond to fitting a Gaussian distribution to the spike-triggered ensemble. If one has access to multi-neuron recordings, a straightforward extension of this approach is to fit a different Gaussian distribution to each binary population pattern. In statistics, the corresponding model is known as the location model [Olkin & Tate 1961, Lauritzen & Wermuth 1989, Krzanowski 1993]. To estimate this model, one has to observe sufficient amounts of data for each population pattern. As the number of possible binary patterns grows exponentially with the number of neurons, it is desirable to include regularization constraints in order to make parameter estimation tractable.

Here, we extend the framework of pairwise maximum entropy modeling to a joint model for binary and continuous variables. This allows us to analyze the functional connection structure in a neural population at the same time as its relationship with further continuous signals of interest. In particular, this approach makes it possible to include a stimulus as a continuous variable into the framework of maximum-entropy modeling. In this way, we can study the stimulus dependence of binary neural population activity in a regularized framework in a rigorous way. In particular, we can use it to extract non-linear features in the stimulus that a population of neurons is sensitive to, while taking the binary nature of spike trains into account.

We discuss the relationship of the obtained features with classical approaches such as spike-triggered average (STA) and spike-triggered covariance (STC). In addition, we show how the model can be used to perform spike-by-spike decoding and yields a natural spike-train metric [Victor & Purpura 1997, Ahmadian *et al.* 2009]. We start with a derivation of the model and a discussion of its features.

5.2 Model formulation

In this section we derive the maximum-entropy model for joint continuous and binary data with second-order constraints and describe its basic properties. We write continuous variables $\mathbf{x}$ and binary variables $\mathbf{b}$. Having observed the joint mean $\boldsymbol{\mu}$ and joint covariance $\mathbf{C}$, we want to find a distribution p_{ME} which achieves the maximal entropy under all distributions with these observed moments. Since we model continuous and binary variables jointly, we define entropy to be a mixed discrete entropy and differential entropy:

$$H[p] = -\sum_{\mathbf{b}} \int p(\mathbf{x}, \mathbf{b}) \log p(\mathbf{x}, \mathbf{b}) \mathrm{d}\mathbf{x}$$

Formally, we require p_{ME} to satisfy the following constraints:

$$\begin{aligned} \mathbb{E}[\mathbf{x}] &= \boldsymbol{\mu}_x \qquad \mathbb{E}[\mathbf{b}] = \boldsymbol{\mu}_b \\ \mathbb{E}[\mathbf{x}\mathbf{x}^\top] &= \mathbf{C}_{xx} + \boldsymbol{\mu}_x \boldsymbol{\mu}_x^\top \qquad \mathbb{E}[\mathbf{b}\mathbf{b}^\top] = \mathbf{C}_{bb} + \boldsymbol{\mu}_b \boldsymbol{\mu}_b^\top \\ \mathbb{E}[\mathbf{x}\mathbf{b}^\top] &= \mathbf{C}_{xb} + \boldsymbol{\mu}_x \boldsymbol{\mu}_b^\top \qquad \mathbb{E}[\mathbf{b}\mathbf{x}^\top] = \mathbf{C}_{bx} + \boldsymbol{\mu}_b \boldsymbol{\mu}_x^\top = \mathbf{C}_{xb} + \boldsymbol{\mu}_x \boldsymbol{\mu}_b^\top \end{aligned} \tag{5.1}$$

where the expectations are taken over p_{ME}. $\mathbf{C}_{xx}, \mathbf{C}_{xb}$ and $\mathbf{C}_{bb}$ are blocks in the observed covariance matrix corresponding to the respective subsets of variables. This problem can be solved analytically using the Lagrange formalism, which leads to a maximum entropy distribution of Boltzmann type:

$$\begin{aligned} p_{\mathrm{ME}}(\mathbf{x}, \mathbf{b} | \boldsymbol{\Lambda}, \boldsymbol{\lambda}) &= \frac{1}{Z(\boldsymbol{\Lambda}, \boldsymbol{\lambda})} \exp\left(Q\left(\mathbf{x}, \mathbf{b} | \boldsymbol{\Lambda}, \boldsymbol{\lambda}\right)\right) \\ Q(\mathbf{x}, \mathbf{b} | \boldsymbol{\Lambda}, \boldsymbol{\lambda}) &= \frac{1}{2} \begin{pmatrix} \mathbf{x} \\ \mathbf{b} \end{pmatrix}^\top \boldsymbol{\Lambda} \begin{pmatrix} \mathbf{x} \\ \mathbf{b} \end{pmatrix} + \boldsymbol{\lambda}^\top \begin{pmatrix} \mathbf{x} \\ \mathbf{b} \end{pmatrix} \\ Z(\boldsymbol{\Lambda}, \boldsymbol{\lambda}) &= \sum_{\mathbf{b}} \int \exp\left(Q\left(\mathbf{x}, \mathbf{b} | \boldsymbol{\Lambda}, \boldsymbol{\lambda}\right)\right) \mathrm{d}\mathbf{x}, \end{aligned} \tag{5.2}$$

where $\boldsymbol{\Lambda}$ and $\boldsymbol{\lambda}$ are chosen such that the resulting distribution fulfills the constraints in equation (5.1), as we discuss below. Before we compute marginal and conditional

distributions in this model, we explore its basic properties. First, we note that the joint distribution can be factorized in the following way:

$$p_{\mathrm{ME}}(\mathbf{x}, \mathbf{b}|\mathbf{\Lambda}, \boldsymbol{\lambda}) = p_{\mathrm{ME}}(\mathbf{x}|\mathbf{b}, \mathbf{\Lambda}, \boldsymbol{\lambda}) p_{\mathrm{ME}}(\mathbf{b}|\mathbf{\Lambda}, \boldsymbol{\lambda}) \tag{5.3}$$

The conditional density $p_{\mathrm{ME}}(\mathbf{x}|\mathbf{b}, \mathbf{\Lambda}, \boldsymbol{\lambda})$ is a Normal distribution, given by:

$$\begin{aligned} p_{\mathrm{ME}}(\mathbf{x}|\mathbf{b}, \mathbf{\Lambda}, \boldsymbol{\lambda}) &\propto \exp\left(\frac{1}{2}\mathbf{x}^\top \mathbf{\Lambda}_{xx}\mathbf{x} + \mathbf{x}^\top (\boldsymbol{\lambda}_x + \mathbf{\Lambda}_{xb}\mathbf{b})\right) \\ &\propto \mathcal{N}\left(\mathbf{x}|\boldsymbol{\mu}_{x|b}, \mathbf{\Sigma}\right) \quad \text{,with} \\ \boldsymbol{\mu}_{x|b} = \mathbf{\Sigma}\left(\boldsymbol{\lambda}_x + \mathbf{\Lambda}_{xb}\mathbf{b}\right), &\qquad \mathbf{\Sigma} = (-\mathbf{\Lambda}_{xx})^{-1} \end{aligned} \tag{5.4}$$

Here, $\mathbf{\Lambda}_{xx}, \mathbf{\Lambda}_{xb}, \mathbf{\Lambda}_{bx}, \boldsymbol{\lambda}_x$ are the blocks in $\mathbf{\Lambda}$ which correspond to $\mathbf{x}$ and $\mathbf{b}$, respectively. While the mean of this Normal distribution dependent on $\mathbf{b}$, the covariance matrix is independent of the specific binary state. The marginal probability $p_{\mathrm{ME}}(\mathbf{b}|\mathbf{\Lambda}, \boldsymbol{\lambda})$ is given by:

$$\begin{aligned} &Z(\mathbf{\Lambda}, \boldsymbol{\lambda}) p_{\mathrm{ME}}(\mathbf{b}|\Lambda, \lambda) \\ &= \exp\left(\frac{1}{2}\mathbf{b}^\top \mathbf{\Lambda}_{bb}\mathbf{b} + \mathbf{b}^\top \boldsymbol{\lambda}_b\right) \int \exp\left(\frac{1}{2}\mathbf{x}^\top \mathbf{\Lambda}_{xx}\mathbf{x} + \mathbf{x}^\top (\boldsymbol{\lambda}_x + \mathbf{\Lambda}_{xb}\mathbf{b})\right) \mathrm{d}\mathbf{x} \\ &= (2\pi)^{\frac{n}{2}} |-\mathbf{\Lambda}_{xx}|^{-\frac{1}{2}} \exp\left(\frac{1}{2}\mathbf{b}^\top \left(\mathbf{\Lambda}_{bb} + \mathbf{\Lambda}_{xb}^\top (-\mathbf{\Lambda}_{xx})^{-1} \mathbf{\Lambda}_{xb}\right)\mathbf{b}\right. \\ &\quad \left. + \mathbf{b}^\top \left(\boldsymbol{\lambda}_b + \mathbf{\Lambda}_{xb}^\top (-\mathbf{\Lambda}_{xx})^{-1} \boldsymbol{\lambda}_x\right) + \frac{1}{2}\boldsymbol{\lambda}_x^\top (-\mathbf{\Lambda}_{xx})^{-1} \boldsymbol{\lambda}_x\right) \end{aligned} \tag{5.5}$$

To evaluate the maximum entropy distribution, we need to compute the partition function, which follows from the previous equation by summing over $\mathbf{b}$:

$$\begin{aligned} Z(\mathbf{\Lambda}, \boldsymbol{\lambda}) = (2\pi)^{\frac{n}{2}} |-\mathbf{\Lambda}_{xx}|^{-\frac{1}{2}} \sum_b \exp\left(\frac{1}{2}\mathbf{b}^\top \left(\mathbf{\Lambda}_{bb} + \mathbf{\Lambda}_{xb}^\top (-\mathbf{\Lambda}_{xx})^{-1} \mathbf{\Lambda}_{xb}\right)\mathbf{b}\right. \\ \left. + \mathbf{b}^\top \left(\boldsymbol{\lambda}_b + \mathbf{\Lambda}_{xb}^\top (-\mathbf{\Lambda}_{xx})^{-1} \boldsymbol{\lambda}_x\right) + \frac{1}{2}\boldsymbol{\lambda}_x^\top (-\mathbf{\Lambda}_{xx})^{-1} \boldsymbol{\lambda}_x\right) \end{aligned} \tag{5.6}$$

Next, we compute the marginal distribution with respect to $\mathbf{x}$. From equation (5.5) and (5.4), we find that $p_{\mathrm{ME}}(\mathbf{x}|\mathbf{\Lambda}, \boldsymbol{\lambda})$ is a mixture of Gaussians, where each Gaussian of equation (5.4) is weighted by the corresponding $p_{\mathrm{ME}}(\mathbf{b}|\mathbf{\Lambda}, \boldsymbol{\lambda})$. While all mixture components have the same covariance, the different weighting terms affect each component's influence on the marginal covariance of $\mathbf{x}$. Finally, we also compute

the conditional density $p_{\mathrm{ME}}(\mathbf{b}|\mathbf{x}, \Lambda, \lambda)$, which is given by:

$$\begin{aligned} p_{\mathrm{ME}}(\mathbf{b}|\mathbf{x}, \boldsymbol{\Lambda}, \boldsymbol{\lambda}) &= \frac{1}{Z'} \exp\left(\frac{1}{2}\mathbf{b}^\top \boldsymbol{\Lambda}_{bb}\mathbf{b} + \mathbf{b}^\top(\boldsymbol{\lambda}_b + \boldsymbol{\Lambda}_{bx}\mathbf{x})\right) \\ Z' &= \sum_{\mathbf{b}} \exp\left(\frac{1}{2}\mathbf{b}^\top \boldsymbol{\Lambda}_{bb}\mathbf{b} + \mathbf{b}^\top(\boldsymbol{\lambda}_b + \boldsymbol{\Lambda}_{bx}\mathbf{x})\right) \end{aligned} \tag{5.7}$$

Note, that the distribution of the binary variables given the continuous variables is again of Boltzmann type.

Parameter fitting To find suitable parameters for given data, we employ a maximum likelihood approach [Ackley *et al.* 1985, MacKay 2003], where we find the optimal parameters via gradient descent:

$$\begin{aligned} l(\boldsymbol{\Lambda}, \boldsymbol{\lambda}) &= \log p(\{\mathbf{x}^{(n)}, \mathbf{b}^{(n)}\}_{n=1}^N | \boldsymbol{\Lambda}, \boldsymbol{\lambda}) \\ &= \sum_n Q(\mathbf{x}^{(n)}, \mathbf{b}^{(n)}|\Lambda, \lambda) - N \log Z(\boldsymbol{\Lambda}, \boldsymbol{\lambda}) \\ \Rightarrow \nabla_{\boldsymbol{\Lambda}} l &= N\left[\left\langle \begin{pmatrix} \mathbf{x} \\ \mathbf{b} \end{pmatrix}\begin{pmatrix} \mathbf{x} \\ \mathbf{b} \end{pmatrix}^\top \right\rangle_{\text{data}} - \left\langle \begin{pmatrix} \mathbf{x} \\ \mathbf{b} \end{pmatrix}\begin{pmatrix} \mathbf{x} \\ \mathbf{b} \end{pmatrix}^\top \right\rangle_{p_{\mathrm{ME}}}\right] \\ \nabla_{\boldsymbol{\lambda}} l &= N\left[\left\langle \begin{pmatrix} \mathbf{x} \\ \mathbf{b} \end{pmatrix} \right\rangle_{\text{data}} - \left\langle \begin{pmatrix} \mathbf{x} \\ \mathbf{b} \end{pmatrix} \right\rangle_{p_{\mathrm{ME}}}\right] \end{aligned} \tag{5.8}$$

To calculate the moments over the model distribution p_{ME} we make use of the above factorization:

$$\begin{aligned} \left\langle \mathbf{x}\mathbf{x}^\top \right\rangle &= \left\langle \left\langle \mathbf{x}\mathbf{x}^\top | \mathbf{b} \right\rangle \right\rangle_b = (-\boldsymbol{\Lambda}_{xx})^{-1} + \left\langle \boldsymbol{\mu}_{x|b}\boldsymbol{\mu}_{x|b}^\top \right\rangle_b \\ \left\langle \mathbf{x}\mathbf{b}^\top \right\rangle &= \left\langle \boldsymbol{\mu}_{x|b}\mathbf{b}^\top \right\rangle_b = \left\langle \mathbf{b}\mathbf{x}^\top \right\rangle^\top, \qquad \langle \mathbf{x} \rangle = \left\langle \boldsymbol{\mu}_{x|b} \right\rangle_b \end{aligned} \tag{5.9}$$

Hence, the only average we actually need to evaluate numerically is the one over the binary variables. Unfortunately, we cannot directly set the parameters for the continuous part, as they depend on the ones for the binary part. However, since the above equations can be evaluated analytically, the difficult part is finding the parameters for the binary variables. In particular, if the number of binary variables is large, calculating the partition function can become infeasible. To some extent, this can be remedied by the use of specialized Monte-Carlo algorithms [Broderick *et al.* 2007].

5.2.1 An illustrative example

In order to gain intuition into the properties of the model, we illustrate it in a simple one-dimensional case. From equation (5.4) for the conditional mean of the continuous variables, we expect the distance between the conditional means $\boldsymbol{\mu}_{x|b}$ to increase with increasing correlation between continuous and binary variables increases. We see that this is indeed the case: While the conditional Gaussians $p(\mathbf{x}|\mathbf{b}=1)$ and $p(\mathbf{x}|\mathbf{b}=0)$ are identical if x and b are uncorrelated (figure 5.1A), a correlation between $\mathbf{x}$ and $\mathbf{b}$ shifts them away from the unconditional mean (figure 5.1B). Also, the weight assigned to each of the two Gaussians can be changed. While in figures 5.1A and 5.1B $\mathbf{b}$ has a symmetric mean of 0.5, a non-symmetric mean leads to an asymmetry in the weighting of each Gaussian illustrated in figure 5.1C.

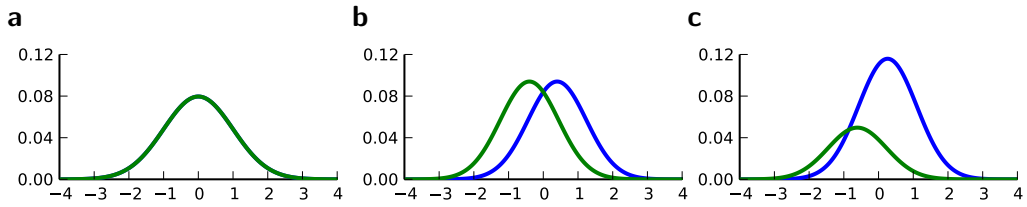

Figure 5.1: Illustration of different parameter settings. **a**:independent binary and continuous variables, **b**: correlations (0.4) between variables, **c**: changing mean of the binary variables (here: 0.7) corresponds to changing weightings of the Gaussians, correlations are 0.4. Blue lines indicate $p(x|b=1)$ and green ones $p(x|b=0)$.

5.2.2 Comparison with other models for the joint modeling of binary and continous data

There are two models in the literature which model the joint distribution of continuous and binary variables, which we will list in the following and compare them to the model derived in this chapter.

Location model The location model (LM) [Olkin & Tate 1961, Lauritzen & Wermuth 1989, Krzanowski 1993] also uses the same factorization as above $p(\mathbf{x},\mathbf{b}) = p(\mathbf{x}|\mathbf{b})p(\mathbf{b})$. However, the distribution for the binary variables $p(\mathbf{b})$ is not of Boltzmann type but a general multinomial distribution and therefore has more degrees of freedom. The conditional distribution $p(\mathbf{x}|\mathbf{b})$ is assumed to be Gaussian with moments $(\boldsymbol{\mu}_\mathbf{b}, \boldsymbol{\Sigma}_\mathbf{b})$, which can both depend on the conditional state $\mathbf{b}$. Thus to fit the LM usually requires much more data to estimate the moments for every possible binary state. The location model can also

be seen as a maximum entropy model in the sense, that it is the distribution with maximal entropy under all distribution with the conditional moments. As fitting this model in its general form is prone to overfitting, various ad hoc constraints have been proposed; see [Krzanowski 1993] for details.

Partially dichotomized Gaussian model Another simple possibility to obtain a joint distribution of continuous and binary variables is to take multivariate (latent) Gaussian distribution for all variables and then dichotomize those components which should represent the binary variables. Thus, a binary variable $\mathbf{b}_i$ is set to 1 if the underlying Gaussian variables is greater than 0 and it is set to 0 if the Gaussian variable is smaller than 0. This model is known as the partially dichotomized Gaussian (PDG) [Cox & Wermuth 1999]. Importantly the marginal distribution over the continuous variables is always Gaussian and not a mixture as in our model. The reason for this is that all marginals of a Gaussian distribution are again Gaussian.

5.3 Applications

5.3.1 Spike triggering and feature extraction

Spike triggering is a common technique in order to find features which a single neuron is sensitive to. The presented model can be seen as an extension in the following sense.

Suppose that we have observed samples $(\mathbf{x}^n, \mathbf{b}^n)^\top$ from a population responding to a stimulus. The spike triggered average (STA) for a neuron i is then defined as

$$\mathrm{STA}_i = \frac{\sum_n \mathbf{x}^n \mathbf{b}_i^n}{\sum_n \mathbf{b}_i^n} = \mathbb{E}[\mathbf{x}\mathbf{b}_i] r_i, \tag{5.10}$$

where $r_i = \frac{\sum_n \mathbf{b}_i^n}{N} = p(\mathbf{b}_i = 1)$ is the firing rate of the i-th neuron or fraction of ones within the sample. Note, that the moment $\mathbb{E}[\mathbf{x}\mathbf{b}_i]$ is one of the constraints we require for the maximum entropy model and therefore the STA is included in the model.

In addition, the model has also similarities to spike-triggered covariance (STC) [Schwartz *et al.* 2002, Pillow & Simoncelli 2006]. STC denotes the distribution or, more precisely, the covariance of the stimuli that evoked a spiking response. Usually, this covariance is then compared to the total covariance over the entire stimulus distribution. In the joint maximum-entropy model, we have access to a similar distribution, namely the conditional distribution $p(\mathbf{x}|\mathbf{b}_i = 1)$, which is a compact

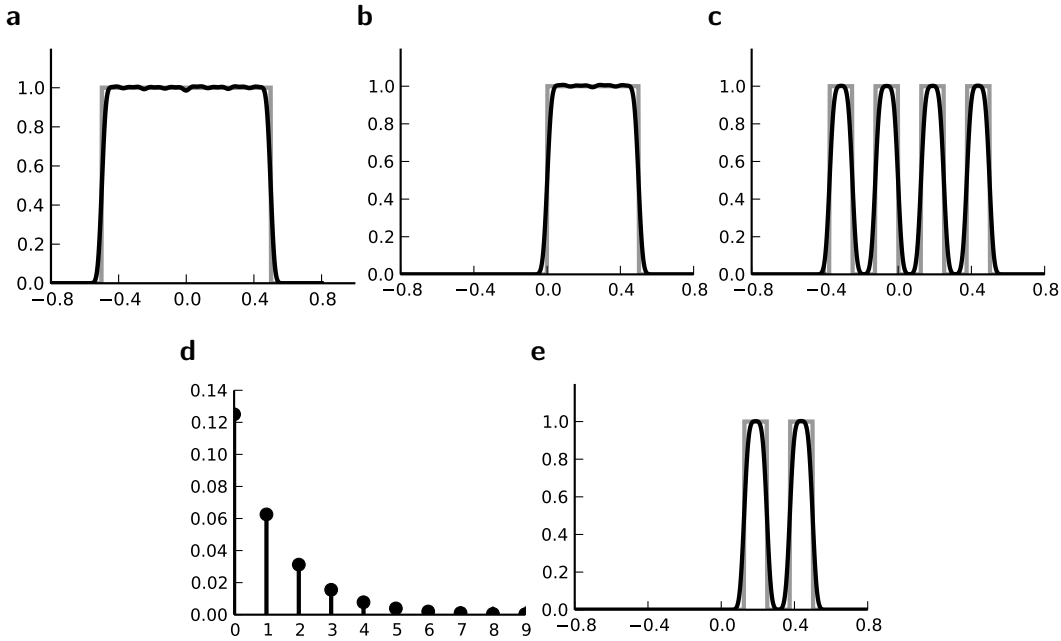

Figure 5.2: Illustration of the binary encoding with box-type tuning curves. **a**: shows the marginal distribution over stimuli. The true underlying stimulus distribution is a uniform distribution over the interval $(-0.5, 0.5)$ and is plotted in shaded gray. The mixture of Gaussian approximation of the MaxEnt model is plotted in black. Each neuron has a tuning-curve, consisting of a superposition of box-functions. **b** shows the tuning-curve of the first neuron. This is equivalent to the conditional distribution, when conditioning on the first bit, which indicates if the stimulus is in the right part of the interval. The tuning-curve is a superposition of 5 box-functions. The true tuning curve is plotted in shaded gray whereas the MaxEnt approximation is plotted in black. **c** shows the tuning curve of neuron with index 2. **d**: Covariance between continuous and binary variables as a function of the index of the binary variables. This is the same as the STA for each neuron (see also equation (5.10)). **e** shows the conditional distribution, when conditioning on both variables (0,2) to be one. This corresponds to the product of the tuning-curves.

description of the spike-triggered distribution. Note that $p(\mathbf{x}|\mathbf{b}_i = 1)$ can be highly non-Gaussian as all neurons $j \neq i$ are marginalized out – this is why the current model is an extension to spike triggering. Additionally, we can also trigger or condition not on a single neuron but on any response pattern $\mathbf{B}_\mathcal{S}$ of a sub-population $\mathcal{S}$. The resulting $p(\mathbf{x}|\mathbf{B}_\mathcal{S})$ with $\mathbf{B}_\mathcal{S} = \{\mathbf{b} : \mathbf{b}_i = \mathbf{B}_i \forall i \in \mathcal{S}\}$ is then also a mixture of Gaussians with 2^n components, where n is the number of unspecified neurons $j \notin \mathcal{S}$. As illustrated above (see figure 5.1B), correlations between neurons and stimuli lead to a separation of the individual Gaussians. Hence, stimulus correlations of other neurons $j \neq i$ in the distribution $p(\mathbf{x}, \mathbf{b}_{j\neq i}|\mathbf{b}_i = 1)$ would have the same effect on the

spike-triggered distribution of neuron i. Correlations within this distribution also imply, that there are correlations between neuron j and neuron i. Thus, stimulus as well as noise correlations cause deviations of the conditional $p(\mathbf{x}|\mathbf{B}_{\mathcal{S}})$ from a single Gaussian. Therefore, the full conditional distribution $p(\mathbf{x}|\mathbf{B}_{\mathcal{S}})$ in general contain more information about the features which trigger this sub-population to evoke the specified response pattern, than the conditional mean, i.e. the STA.

We demonstrate the capabilities of this approach by considering the following encoding. As stimulus, we consider one continuous real valued variable that is drawn uniformly from the interval $[-0.5, 0.5]$. It is mapped to a binary population response in the following way. Each neuron i has a square-wave tuning function:

$$\mathbf{b}_i(\mathbf{x}) = \Theta\left(\sin\left(2\pi(i+1)\mathbf{x}\right)\right),$$

where Θ is the Heaviside function. In this way, the response of a neuron is set to 1 if its tuning-function is positive and 0 otherwise. The first (index 0) neuron distinguishes the left and the right part of the entire interval. The $(i+1)$st neuron distinguishes subsequently left from right in the sub-intervals of the ith neuron. That is, the response of the second neuron is always 1, if the stimulus is in the right part of the intervals $[-0.5, 0]$ and $[0, 0.5]$. These tuning curves can also be thought of as a mapping into a non-linear feature space in which the neuron acts linear again. Although the data-generation process is not contained in our model class we were able to extract the tuning curves as shown in figure 5.2. Note, that for this example neither the STA nor STC analysis alone would provide any insight into the feature selectivity of the neurons, in particular for the neurons which have multi-modal tuning curves (the ones with higher indexes in the above example). However, the tuning curves could be reconstructed with any kind of density estimation, given the STA.

5.3.2 Spike-by-spike decoding

Since we have a simple expression for the conditional distribution $p(\mathbf{x}|\mathbf{b}, \Lambda, \lambda)$ (see equation (5.4)), we can use the model to analyze the decoding performance of a neural population. To illustrate this, we sampled spike trains from two leaky integrate-and-fire neurons for 1 second and discretized the resulting spike trains into 5 bins of 200 ms length each. Each trial, we used a constant two dimensional stimulus, which was drawn from two independent Gamma distributions with shape parameter $\alpha = 3$ and scale parameter $\beta = 0.3$. For each LIF neuron, this two dimensional stimulus was then projected onto the one-dimensional subspace spanned by its receptive field and used as input current. Hence, there are 10 binary variables,

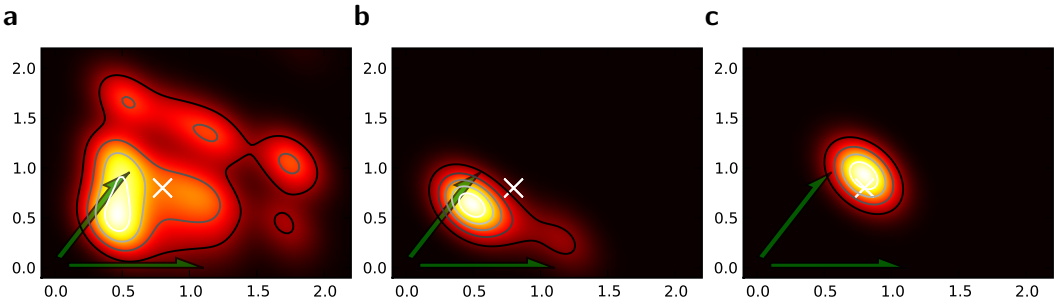

Figure 5.3: Illustration of a spike-by-spike decoding scheme. The MaxEnt model was fit to data from two deterministic integrate-and-fire models. The MaxEnt model can then be used for decoding spikes generated by the two independent deterministic models. The two green arrows correspond the weights of a two-pixel receptive field for each of the two neurons. The 2 dimensional stimulus was drawn from two independent Gamma distributions. The resulting spike-trains were discretized in 5 time-bins, each 200 ms long. A spike-train to a particular stimulus ($\mathbf{x}^\dagger$ cross) is decoded. In **a**) the marginal distribution of the continuous variables is shown. In **b**) the posterior, when conditioning on the first temporal half of the response to that stimulus is shown. Finally in **c**) the conditional distribution, when conditioning on the full observed binary pattern is plotted.

5 for each spike-train of the neurons and 2 continuous variables for the stimulus to be modeled. We draw $5 \cdot 10^6$ samples, calculated the second order moments of the joint stimulus and response vectors and fitted our maximum entropy model to these moments. The obtained distribution is shown in figure 5.3. In 5.3A, we show the marginal distribution of the stimuli, which is a mixture of 2^{10} Gaussians. The receptive fields of the two neurons are indicated by green arrows. To illustrate the decoding process, we sampled a stimulus and corresponding response r, from which we try to reconstruct the stimulus. In 5.3B, we show the conditional distribution when conditioning on the first half of the response. Finally in 5.3C, the complete posterior is shown when conditioned on the full response. From a-c, the posterior is more and more concentrated around the true stimulus. Although there is no neural noise in the encoding model, the reconstruction is not perfect. This is due to the regularization properties of the maximum entropy approach.

5.3.3 Stimulus dependence of firing patterns

While previous studies on the structure of neuronal firing patterns in the retina have compared how well second-order maximum entropy models fit the empirically observed distributions under different stimulation conditions [Schneidman *et al.* 2006, Shlens *et al.* 2006], the stimulus has never been explicitly taken into account into

the model. In the proposed framework, we have access to $p(\mathbf{b}|\mathbf{x})$, so we can explicitly study how the pattern distribution of a neural population depends on the stimulus. We illustrate this by continuing the example of figure 5.3. First, we show how the individual firing probabilities depend on x (figure 5.4A). Note, that although the encoding process for the previous example was noiseless, that is, for every given stimulus there is only one response pattern, the conditional distribution $p(\mathbf{b}|\mathbf{x})$ is not a delta-function, but dispersed around the expected response. This is due to the second order approximation to the encoding model. Further, as it turns out, that a spike in the next bin after a spike is very unlikely under the model, which captures the property of the leaky integrator. Also, we compare how $p(\mathbf{b}|\mathbf{x})$ changes for different values of $\mathbf{x}$. This is illustrated in figure 5.4B.

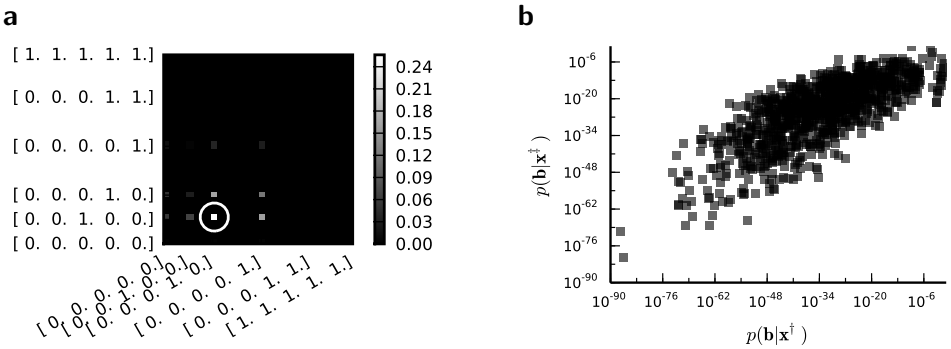

Figure 5.4: Illustration of the conditional probability $p(\mathbf{b}|\mathbf{x})$ for the example in figure 5.3. In 5.4A, for every binary pattern the corresponding probability is plotted for the given stimulus from figure 5.3, where the brightness of each square indicates its probability. For the given stimulus the actual response pattern used for figure 5.3 is marked with a circle. Each pattern $\mathbf{b}$ is split into two halves by the contributions of the two neurons (32 possible patterns for each neuron) and response patterns of the first neuron are shown on the x-axis, while response patterns of the second neuron on the y-axis. In 5.4B we plotted for each pattern $\mathbf{b}$ its probability under the two conditional distributions $p(\mathbf{b}|\mathbf{x}^\dagger)$ and $p(\mathbf{b}|\mathbf{x}^\ddagger)$ against each other with $\mathbf{x}^\dagger = (0.85, 0.72)$ and $\mathbf{x}^\ddagger = (1.5, 1.5)$.

5.3.4 A spike train metric

Oftentimes, it is desirable to measure distances between spike trains [Victor & Purpura 1997]. One problem, however, is that not every spike might be of equal importance. That is, if a spike train differs only in one spike, it might nevertheless represent a completely different stimulus. Therefore, Ahmadian [Ahmadian *et al.* 2009] suggested to measure the distance between spike trains as the difference of stimuli when reconstructed based on the one or the other spike train

seems. If the population is noisy, we want to measure the difference of reconstructed stimuli on average. To this end, we need access to the posterior distribution, when conditioning on a particular spike train or binary pattern. Using the maximum entropy model, we can define the following spike-metric:

$$\begin{aligned} d(\mathbf{b}^1, \mathbf{b}^2) &= D_{\mathrm{KL}}\left[p_{\mathrm{ME}}(\mathbf{x}|\mathbf{b}^1)||p_{\mathrm{ME}}(\mathbf{x}|\mathbf{b}^2)\right] \\ &= \frac{1}{2}\left(\left(\boldsymbol{\mu}_{x|b^1} - \boldsymbol{\mu}_{x|b^2}\right)^\top \boldsymbol{\Lambda}_{xx}\left(\boldsymbol{\mu}_{x|b^1} - \boldsymbol{\mu}_{x|b^2}\right)\right) \end{aligned} \tag{5.11}$$

Here, D_{KL} denotes the Kullback-Leibler divergence between the posterior densities. Equation 5.11 is symmetric in $\mathbf{b}$, however, in order to get a symmetric expression for other types of posterior distributions, the Jensen-Shannon divergence might be used instead. As an example we consider the induced metrics for the encoding model of figure 5.2. The metric induced by the square-wave tuning functions of section 5.3.1 is relatively simple. When conditioning on a particular population response, the conditional distribution $p(\mathbf{x}|\mathbf{b})$ is always a Gaussian with approximately the width of the smallest wavelength. Flipping a neuron's response within this pattern corresponds to shifting the conditional distribution. Suppose we have observed a population response consisting of only ones. This results in a Gaussian posterior distribution with mean in the middle of the rightmost interval $(0.5 - \frac{1}{1024}, 0.5)$. Now flipping the response of the "low-frequency" neuron, that is the one shown in figure 5.2B, shifts the mean of the posterior to the middle of the sub-interval $(-\frac{1}{1024}, 0)$. Whereas flipping the "high-frequency" neuron, the one which indicates left or right within the smallest possible sub-interval, corresponds to shifting the mean just by the amount of this smallest interval to the left. Flipping the response of single neurons within this population can result in posterior distribution which look quite different in terms of the Kullback-Leibler divergence. In particular, there is an ordering in terms of the frequency of the neurons with respect to the proposed metric.

5.4 Discussion

We have presented a maximum-entropy model based on the joint second order statistics of continuous valued variables and binary neural responses. This allows us to extend the maximum-entropy approach [Schneidman *et al.* 2006] for analyzing neural data to incorporate other variables of interest such as continuous valued stimuli. Alternatively, additional neurophysiological signals such as local field potentials [Montemurro *et al.* 2008] can be taken into account to study their relation with the joint firing patterns of local neural ensembles. We have demonstrated

four applications of this approach: (1) It allows us to extract the features a (sub-)population of neurons is sensitive to, (2) we can use it for spike-by-spike decoding, (3) we can assess the impact of stimuli on the distribution of population patterns and (4) it yields a natural spike-train metric.

We have shown that the joint maximum-entropy model can be learned in a convex fashion, although high-dimensional binary patterns might require the use of efficient sampling techniques. Because of the maximum-entropy approach the resulting distribution is well regularized and does not require any ad-hoc restrictions or regularity assumptions as have been proposed for related models [Krzanowski 1993]. Analogous to a Boltzmann machine with hidden variables, it is possible to further add hidden binary nodes to the model. This allows us to take higher-order correlations into account as well, although we stay essentially in the second-order framework. Fortunately, the learning scheme for fitting the modified model to observed data remains almost unchanged: The only difference is that the moments have to be averaged over the non-observed binary variables as well. In this way, the model can also be used as a clustering algorithm if we marginalize over all binary variables. The resulting mixture of Gaussian model will consist of 2^N components, where N is the number of hidden binary variables. Unfortunately, convexity cannot be guaranteed if the model contains hidden nodes. In a similar fashion, we could also add hidden continuous variables, for example to model unobserved common inputs. In contrast to hidden binary nodes, this does not lead to an increased model complexity: averaging over hidden continuous variables corresponds to integrating out each Gaussian within the mixture, which results in another Gaussian. Also the restriction that all covariance matrices in the mixture need to be the same still holds, because each Gaussian is integrated in the same way.

6

Conclusion

In this thesis, we have presented several methods for dealing with certain aspects of neural coding. This includes the encoding, the decoding and joint modeling of spikes and stimuli. In each of these subtasks we have consistently followed a Bayesian approach which explicitly models not only the uncertainty over the data but also about parameters which specify the generating model. Specifically, we have presented methods for Bayesian system identification in chapter 3 and for decoding of stimuli from spikes in chapter 4. Finally, in chapter 5 we presented a model based on the maximum entropy principle to estimate the joint probability of spikes and stimuli. In practice, exact computation of the involved distributions is often intractable. Therefore, we had to develop approximation schemes to overcome this obstacle. As the mapping from stimuli to spikes or vice versa is likely to be complex, we think a Bayesian treatment is of key importance as it provides a principled way of controlling the complexity of a model in situations where the parameters are underconstrained by the data. In addition, a quantitative description of he uncertainty is crucial for rigorous model comparison. Therefore, we believe that the Bayesian approach to neural coding developed in this thesis improves the foundations for a quantitative analysis of the neural code.

A

Appendix

A.1 Expectation Propagation with Gaussians

In the following we will explain the essentials for approximating posterior distributions with a Gaussian distribution via the Expectation Propagation algorithm.

Suppose the joint distribution of a parameter vector of interest $\mathbf{w}$ and n independent observations $D = \{x_1, \ldots, x_n\}$ factors as:

$$p(D, \mathbf{w}) = p(\mathbf{w}) \prod_{i=1}^{n} p(x_i|\mathbf{w}), \tag{A.1}$$

where $p(\mathbf{w})$ is a chosen prior distribution. Further we assume, that each of the likelihood factors depends on a linear projection of the parameters $\mathbf{w}$ only. That is a likelihood factor can be written as

$$p(x_i|\mathbf{w}) = p(x_i|\boldsymbol{\psi}_i^\top \mathbf{w}). \tag{A.2}$$

Hence, each likelihood factor is intrinsically one-dimensional. Next, we choose an (un-normalized) Gaussian $\tilde{t}_i$ with which we would like to approximate each of those

factors:

$$p(x_i|\boldsymbol{\psi}_i^\top \mathbf{w}) \approx \exp\left(-\frac{1}{2}\pi_i \left(\boldsymbol{\psi}_i^\top \mathbf{w}\right)^2 + b_i \left(\boldsymbol{\psi}_i^\top \mathbf{w}\right)\right) \tag{A.3}$$

$$= \exp\left(-\frac{1}{2}\pi_i \mathbf{w}^\top \left(\boldsymbol{\psi}_i \boldsymbol{\psi}_i^\top\right) \mathbf{w} + b_i \mathbf{w}^\top \left(\boldsymbol{\psi}_i\right)\right) =: \tilde{t}_i(\boldsymbol{\psi}_i^\top \mathbf{w}) \tag{A.4}$$

Plugging this into equation (A.1), we obtain for the approximation $Q(\mathbf{w}|D)$ to the posterior:

$$Q(\mathbf{w}|D) = \exp\left(-\frac{1}{2}\mathbf{w}^\top \left(\sum_i^n \pi_i \boldsymbol{\psi}_i \boldsymbol{\psi}_i^\top\right) \mathbf{w} + \mathbf{w}^\top \left(\sum_i^n b_i \boldsymbol{\psi}_i\right)\right) p(\mathbf{w}) \tag{A.5}$$

The prior distribution $p(\mathbf{w})$ is allowed to have two different forms. It can either be a Gaussian in which case the inverse prior covariance has to be added to the outer products of the features $\boldsymbol{\psi}_i$. Another option is, that the prior distribution also factorizes into intrinsic one-dimensional terms. This would be the case for example, if a Laplace prior is used.

$$\begin{aligned} p(\mathbf{w}) &\propto \prod_k \exp\left(-\tau|\mathbf{w}_k|\right) \\ &= \prod_k p_p\left(\boldsymbol{\psi}_k^\top \mathbf{w}\right) \\ &\text{with} \\ p_p(u) = \exp(|u|), &\qquad \boldsymbol{\psi}_k = (0, \dots, 0, \underbrace{1}_{k}, 0, \dots)^\top \end{aligned} \tag{A.6}$$

In order to obtain the desired Gaussian approximation to the true posterior, the problem is now to find the parameters π_i, b_i. Once these parameters are found, we get the desired approximation via equation (A.1). If the posterior consists of a single factor, then the desired parameters π_1, b_1 are easily obtained via moment matching. The moments usually have to be calculated by a numerical one-dimensional integration along the direction $\boldsymbol{\psi}_1$. To incorporate a new factor, we fix the parameters of the first one and try to find suitable b_2, π_2 for the second factor. More precisely, we want to minimize the Kullback-Leibler distance:

$$D_{\mathrm{KL}}\left[Q(\mathbf{w}|\{x_1, x_2\})||Q(\mathbf{w}|\{x_1\})p(x_2|\boldsymbol{\psi}_2^\top \mathbf{w})\right] \tag{A.7}$$

$$= D_{\mathrm{KL}}\left[Q(\mathbf{w}|\{x_1\}) \exp\left(-\frac{1}{2}\pi_2 \left(\boldsymbol{\psi}_2^\top \mathbf{w}\right)^2 + b_2(\boldsymbol{\psi}_2^\top \mathbf{w})\right) ||Q(\mathbf{w}|\{x_1\})p(x_2|\boldsymbol{\psi}_2^\top \mathbf{w})\right] \tag{A.8}$$

As both Q distributions are the same and all other factors vary only along one

dimension $\boldsymbol{\psi}_2$, the only degree of freedom we have are the moments in that direction (see [Seeger 2005]). Technical speaking, we can split the integration of the Kullback-Leibler distance into two parts. One over the direction $\boldsymbol{\psi}_2$ and one in the orthogonal direction. Now, for notational simplicity, we denote $\boldsymbol{\psi}_2^\top \mathbf{w} =: u_2$. The moments of the Gaussian side in equation (A.8) can easily be computed by looking at the exponent. Let μ_1, σ_1 be the moments of the Q distribution in the direction of $\boldsymbol{\psi}_2$:

$$-\frac{1}{2\sigma_1}(u_2-\mu_1)^2 - \frac{1}{2}\pi_2 u_2^2 + b_2 u_2 \tag{A.9}$$

$$= -\frac{1}{2}u_2^2\left(\frac{1}{\sigma_1}+\pi_2\right) + u_2\left(\frac{\mu_1}{\sigma_1}+b_2\right) - \frac{1}{2}\frac{\mu_1^2}{\sigma_1} \tag{A.10}$$

Thus the moments μ_2, σ_2 are:

$$\sigma_2 = \left(\frac{1}{\sigma_1}+\pi_2\right)^{-1} \tag{A.11}$$

$$\mu_2 = \sigma_2\left(\frac{\mu_1}{\sigma_1}+b_2\right) \tag{A.12}$$

Now, these moments have to be matched with the numerically obtained ones μ_2', σ_2' of $Q(u_2|\{x_1\})p(x_2|u_2)$ by adjusting π_2, b_2. This can be done, by choosing the parameters according to:

$$\pi_2 = \frac{1}{\sigma_2'} - \frac{1}{\sigma_1} \tag{A.13}$$

$$b_2 = \mu_2'\left(\frac{1}{\sigma_1}+\pi_2\right) - \frac{\mu_1}{\sigma_1} \tag{A.14}$$

In this fashion we can incorporate one likelihood factor after another. This procedure is known as assumed density filtering (see [Minka 2001]). The obtained approximation to the posterior depends on the order in which we incorporate the likelihood factors. The idea of Expectation Propagation is not to stop after one such sweep over the factors. EP rather tries to fulfill the consistency [Opper & Winther 2005]:

$$\frac{Q(\mathbf{w}|\{x_1,\ldots,x_n\})}{\exp\left(-\frac{1}{2}\pi_i\left(\boldsymbol{\psi}_i^\top\mathbf{w}\right)^2 + b_i\left(\boldsymbol{\psi}_i^\top\mathbf{w}\right)\right)} p(x_i|\boldsymbol{\psi}_i^\top\mathbf{w}) \overset{D_{\mathrm{KL}}}{=} Q(\mathbf{w}|\{x_1,\ldots,x_n\}) \tag{A.15}$$

That is, we replace one of the approximating factors with the original one and require the moments not to change. To achieve this, one usually select an arbitrary factor i and divide it out of the current approximation. The resulting distribution is called the cavity distribution $Q^{\setminus i}(\mathbf{w})$. If we call the current moments of the

approximation $\boldsymbol{\mu}, \boldsymbol{\Sigma}$, the moments in the direction of $\boldsymbol{\psi}_i$ are given by:

$$\mu_i = \boldsymbol{\psi}_i^\top \boldsymbol{\mu} \tag{A.16}$$

$$\sigma_i = \boldsymbol{\psi}_i^\top \boldsymbol{\Sigma} \boldsymbol{\psi}_i \tag{A.17}$$

Thus, we have for the cavity distribution:

$$Q^{\setminus i}(\boldsymbol{\psi}_i^\top \mathbf{w}) = \frac{Q(\boldsymbol{\psi}_i^\top \mathbf{w} | \{x_1, \ldots, x_n\})}{\exp\left(-\frac{1}{2}\pi_i \left(\boldsymbol{\psi}_i^\top \mathbf{w}\right)^2 + b_i \left(\boldsymbol{\psi}_i^\top \mathbf{w}\right)\right)} \tag{A.18}$$

$$= \exp\left(-\frac{1}{2}\frac{(u_i - \mu_i)^2}{\sigma_i} + \frac{1}{2}\pi_i u_i^2 - b_i u_i\right) \tag{A.19}$$

Where we have abbreviated $u_i := \boldsymbol{\psi}_i^\top \mathbf{w}$. By using the same algebra as before, we have for the moments of the cavity distribution:

$$\sigma_i^{\setminus i} = \left(\frac{1}{\sigma_i} - \pi_i\right)^{-1} \tag{A.20}$$

$$\mu_i^{\setminus i} = \sigma_i^{\setminus i}\left(\frac{\mu_i}{\sigma_i} - b_i\right) \tag{A.21}$$

Now, we are in the same situation as before, because we want to update the parameters π_i, b_i in order to match the moments of the approximation to the ones of the cavity distribution times the original factor. These moments have to be calculated numerically, which can efficiently be computed as the involved integrals are only one-dimensional. We call these numerical moments μ_i', σ_i':

$$\mathbb{E}_{Q^{\setminus i}(u_i)p(x_i|u_i)}\left[u_i\right] = \mu_i' \tag{A.22}$$

$$\mathbb{E}_{Q^{\setminus i}(u_i)p(x_i|u_i)}\left[(u_i - \mu_i)^2\right] = \sigma_i' \tag{A.23}$$

The moments have to match those of the complete approximation which gives:

$$\sigma_i' \stackrel{!}{=} \left(\frac{1}{\sigma_i^{\setminus i}} + \pi_i^{\text{new}}\right)^{-1} \tag{A.24}$$

$$\mu_i' \stackrel{!}{=} \left(\frac{1}{\sigma_i^{\setminus i}} + \pi_i^{\text{new}}\right)\left(\frac{\mu_i^{\setminus i}}{\sigma_i^{\setminus i}} + b_i^{\text{new}}\right) \tag{A.25}$$

$$\Rightarrow \pi_i^{\text{new}} = \frac{1}{\sigma_i'} - \frac{1}{\sigma_i^{\setminus i}} \tag{A.26}$$

$$b_i^{\text{new}} = \mu_i'\left(\frac{1}{\sigma_i^{\setminus i}} + \pi_i^{\text{new}}\right) - \frac{\mu_i^{\setminus i}}{\sigma_i^{\setminus i}} \tag{A.27}$$

Now we can plug in the definition of the moments of the cavity distribution to get an update for the parameters:

$$\Delta\pi_i = \pi^{\text{new}} - \pi^{\text{old}} \tag{A.28}$$

$$\Delta b_i = b^{\text{new}} - b^{\text{old}} \tag{A.29}$$

Together with equation (A.5) this results in a rank one update of the full distribution over the complete parameter vector $\mathbf{w}$. More precisely we have a rank one update of the covariance matrix of the approximating Gaussian as well as an update of the mean:

$$\begin{aligned} \boldsymbol{\Sigma}^{\text{new}} &= \boldsymbol{\Sigma}^{\text{old}} - \boldsymbol{\psi}_i \boldsymbol{\psi}_i^\top \frac{\Delta\pi_i}{1+\sigma_i\Delta\pi_i} \\ \boldsymbol{\mu}^{\text{new}} &= \boldsymbol{\mu}^{\text{old}} + \frac{\Delta b_i - \mu_i\Delta\pi_i}{1+\sigma_i\Delta\pi_i}\boldsymbol{\psi}_i \end{aligned} \tag{A.30}$$

Where we have used the Woodbury identity to obtain equation (A.30). To implement these equations in a numerically stable manner, one usually represents the covariance by it Cholesky decomposition:

$$\boldsymbol{\Sigma} = \mathbf{L}\mathbf{L}^\top, \tag{A.31}$$

where $\mathbf{L}$ is a lower triangular matrix. To calculate the moments for the Laplace factors, we used a technique by [Seeger 2008] as numerical integration of Laplace factors can unstable.

Marginal likelihood The marginal Likelihood for the hyperparameters θ is defined by:

$$\begin{aligned} L(\theta, \text{Model}) &= P(D|\theta, \text{Model}) \\ &= \int P(D, \mathbf{w}|\theta, \text{Model})\text{d}\mathbf{w} \\ &= \int P(\mathbf{w}|\theta, \text{Model}) \prod_{i=1}^{n} P(x_i|\mathbf{w}, \theta, \text{Model})\text{d}\mathbf{w} \end{aligned} \tag{A.32}$$

When considering only the parameters π_i, b_i, EP gives us an un-normalized approximation to the likelihood factors $\tilde{t}_i(\mathbf{w})$. As long as one is interested in the posterior

only, this does not matter, because:

$$P(\mathbf{w}|D) = \frac{P(\mathbf{w}|\theta, \text{Model}) \prod_{i=1}^{n} \tilde{t}_i(\mathbf{w}) C_i}{\int P(\mathbf{w}|\theta, \text{Model}) \prod_{i=1}^{n} \tilde{t}_i(\mathbf{w}) C_i \mathrm{d}\mathbf{w}} = \frac{P(\mathbf{w}|\theta, \text{Model}) \prod_{i=1}^{n} \tilde{t}_i(\mathbf{w})}{\int P(\mathbf{w}|\theta, \text{Model}) \prod_{i=1}^{n} \tilde{t}_i(\mathbf{w}) \mathrm{d}\mathbf{w}} \tag{A.33}$$

However, if we want to approximate the marginal likelihood we need the C_i explicitly:

$$L(\theta, \text{Model}) \approx \int P(\mathbf{w}|\theta, \text{Model}) \prod_{i=1}^{n} C_i \tilde{t}_i(\mathbf{w}|\theta, \text{Model}) \mathrm{d}\mathbf{w} \tag{A.34}$$

The idea is to not only match the moments but the 0th moments as well. We require the expectation of $P(x_i|\mathbf{w})$ and $\tilde{t}_i(\mathbf{w})$ under $Q^{\setminus i}(\mathbf{w})$ to be the same for all i, from which we obtain:

$$\begin{aligned} Z_i := E_{Q^{\setminus i}}[P(x_i|\mathbf{w})] &= E_{Q^{\setminus i}}[C_i \tilde{t}_i(\mathbf{w})] \\ &= C_i \underbrace{E_{Q^{\setminus i}}[\tilde{t}_i(\mathbf{w})]}_{=:\tilde{Z}_i} \end{aligned} \tag{A.35}$$

For the $\tilde{Z}_i$ we have:

$$\begin{aligned} \tilde{Z}_i &= \frac{1}{\sqrt{2\pi\sigma_{\setminus i}}} \int \exp(-\frac{1}{2}\pi_i u_i^2 + b_i u_i) \exp\left(-\frac{1}{2}\frac{(u_i - \mu_{\setminus i})^2}{\sigma_{\setminus i}}\right) \mathrm{d}u_i \\ &= \frac{1}{\sqrt{2\pi\sigma_{\setminus i}}} \int \exp\left(-\frac{1}{2}\frac{\left(u_i - \left(\pi_i + \sigma_{\setminus i}^{-1}\right)^{-1}\left(b_i + \sigma_{\setminus i}^{-1}\mu_{\setminus i}\right)\right)^2}{\left(\pi_i + \sigma_{\setminus i}^{-1}\right)^{-1}}\right) \mathrm{d}u_i \\ &\qquad \cdot \exp\left(-\frac{1}{2}\mu_{\setminus i}^2 {\sigma_{\setminus i}}^{-1} + \frac{1}{2}\left(\pi_i + \sigma_{\setminus i}^{-1}\right)^{-1}\left(b_i + \sigma_{\setminus i}^{-1}\mu_{\setminus i}\right)^2\right) \\ &= \frac{\sqrt{2\pi(\pi_i + \sigma_{\setminus i}^{-1})}}{\sqrt{2\pi\sigma_{\setminus i}}} \exp\left(-\frac{1}{2}\frac{\left(\sigma_{\setminus i} b_i^2 + 2\mu_{\setminus i} b_i - \pi_i \mu_{\setminus i}^2\right)}{\pi_i \sigma_{\setminus i} + 1}\right) \end{aligned} \tag{A.36}$$

Therefore, we have for the marginal likelihood:

$$
\begin{aligned}
\log C_i &= \log Z_i - \log \tilde{Z}_i \\
\Rightarrow \log L(\theta, \text{Model}) &= \log \int \exp\left(\sum_{i=1}^{n} \log C_i - \frac{1}{2}\pi_i \mathbf{w}^\top \boldsymbol{\psi}_i \boldsymbol{\psi}_i^\top \mathbf{w} + b_i \boldsymbol{\psi}_i^\top \mathbf{w}\right) \mathrm{d}\mathbf{w} \\
&= \sum_{i=1}^{n} \log C_i + (2\pi)^{\frac{D}{2}} |\boldsymbol{\Sigma}_p|^{\frac{1}{2}} \exp\left(\frac{1}{2}\boldsymbol{\mu}_p^\top \boldsymbol{\Sigma}_p^{-1} \boldsymbol{\mu}_p\right) \\
&\text{where} \\
\boldsymbol{\Sigma}_p &= \left(\sum_i \pi_i \boldsymbol{\psi}_i \boldsymbol{\psi}_i^\top\right)^{-1} \\
\boldsymbol{\mu}_p &= \boldsymbol{\Sigma}_p \left(\sum_i b_i \boldsymbol{\psi}_i\right)
\end{aligned}
\tag{A.37}
$$

One can also calculate gradients of the marginal likelihood with respect to hyper parameters (see [Seeger 2005]).

A.2 Bayes-optimal point estimate for average log-loss

In the following we consider a simple example of a coin flip to illustrate the potential benefit of an optimized point estimate for the expected loss after having observed the data. Let x be Bernoulli distributed with unkown parameter $\theta \in [0, 1]$. If we observe N data points $x_i \in \{0, 1\}$ with k ones and assume a uniform prior over $\theta \sim U[0, 1]$, we can compute the posterior distribution for θ:

$$
\begin{aligned}
p(\theta|\{x_i\}) &= \frac{1}{Z} \prod_i \theta^{x_i} (1-\theta)^{1-x_i} \\
Z &= \int_0^1 \prod_i \theta^{x_i} (1-\theta)^{1-x_i} \, \mathrm{d}\theta,
\end{aligned}
$$

which is a Beta-distribution with parameters $\alpha = k + 1, \beta = N + 1$. The posterior mean is given by $\mu = \frac{k+1}{N+2}$. We define the average log-loss to be:

$$
\text{loss}(\theta, \hat{\theta}) = \sum_{x=0,1} -p(x|\theta) \log p(x|\hat{\theta})
$$

Then, we can calculate the expected average log-loss after having observed the data $\{x_i\}$:

$$\begin{aligned} F(\hat{\theta}) &:= \int \left[\sum_{x=0,1} -p(x|\theta) \log p(x|\hat{\theta}) \right] p(\theta|\{x_i\}) \mathrm{d}\theta \\ &= \int \Big[-\log(\hat{\theta})\theta - \log(1-\hat{\theta})(1-\theta) \Big] p(\theta|\{x_i\}) \mathrm{d}\theta \\ &= -\mu \log(\hat{\theta}) - (1-\mu)\log(1-\hat{\theta}) \end{aligned}$$

F can now be minimized with respect to the point-estimate $\hat{\theta}$. The derivative with respect to $\hat{\theta}$ is given by:

$$\begin{aligned} 0 \stackrel{!}{=} \frac{\mathrm{d}F}{\mathrm{d}\hat{\theta}} &= -\frac{\mu}{\hat{\theta}} + \frac{1-\mu}{1-\hat{\theta}} \\ \Rightarrow \hat{\theta} &= \mu \end{aligned}$$

Therefore the posterior mean optimizes the expected prediction performance as measured by the average log-loss. We can also calculate the difference in expected performance between the posterior mean and the MAP, which is given by $\theta_{\mathrm{MAP}} = \frac{k}{N}$. The difference in expected performance is given by:

$$\begin{aligned} F(\theta_{\mathrm{MAP}}) - F(\mu) &= -\mu \log(\theta_{\mathrm{MAP}}) - (1-\mu)\log(1-\theta_{\mathrm{MAP}}) + \mu\log(\mu)(1-\mu)\log(1-\mu) \\ &= \mu \log\left(\frac{\mu}{\theta_{\mathrm{MAP}}}\right) + (1-\mu)\log\left(\frac{1-\mu}{1-\theta_{\mathrm{MAP}}}\right) \end{aligned}$$

The difference in expected log-loss is the Kullback-Leibler divergence between the distribution corresponding to the optimized estimate (the posterior mean) and the distribution induced by the MAP estimate. As the Kullback-Leibler divergence is always nonnegative, this shows that the loss incurred by the MAP estimate is greater than the optimized estimate, irrespective of the data (k) that was observed. In the extreme cases, i.e. $k = 0$ or $k = N$, the difference becomes infinite. This simple example shows that, in principle, an extra gain in performance can be achieved by optimizing the parameters for the expected performance over the posterior distribution.

Bibliography

[Abbott & Dayan 1999] LF Abbott and P. Dayan. *The Effect of Correlated Variability on the Accuracy of a Population Code.* Neural Computation, vol. 11, no. 1, pages 91–101, 1999. 62

[Ackley *et al.* 1985] D.H. Ackley, G.E. Hinton and T.J. Sejnowski. *A learning algorithm for Boltzmann machines.* Cognitive Science, vol. 9, pages 147–169, 1985. 97

[Ahmadian *et al.* 2009] Y. Ahmadian, J. Pillow, J. Shlens, E. Simoncelli, E.J. Chichilinsky and L. Paninski. *A decoder-based spike train metric for analyzing the neural code in the retina.* In Frontiers in Systems Neuroscience. Conference Abstract: Computational and systems neuroscience, 2009. 95, 103

[Allen 2007] E. Allen. Modeling with Itô Stochastic Differential Equations: theory and applications. Springer Verlag, 2007. 11

[Andrew & Gao 2007] G. Andrew and J. Gao. *Scalable training of L 1-regularized log-linear models.* In Proceedings of the 24th International Conference on Machine learning, page 40. ACM, 2007. 29

[Arcas & Fairhall 2003] B.A. Arcas and A.L. Fairhall. *What causes a neuron to spike?* Neural Computation, vol. 15, no. 8, pages 1789–1807, 2003. 65, 69

[Barbieri *et al.* 2001] R. Barbieri, M.C. Quirk, L.M. Frank, M.A. Wilson and E.N. Brown. *Construction and analysis of non-Poisson stimulus-response models of neural spiking activity.* Journal of Neuroscience Methods, vol. 105, no. 1, pages 25–37, 2001. 6

[Bell & Sejnowski 1995] A.J. Bell and T.J. Sejnowski. *An Information-Maximization Approach to Blind Separation and Blind Deconvolution.* Neural Computation, vol. 7, no. 6, pages 1129–1159, 1995. 90

[Berens *et al.* 2009] Philipp Berens, Sebastian Gerwinn, Alexander Ecker and Matthias Bethge. *Neurometric function analysis of population codes.* In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams and A. Culotta, editeurs, Advances in Neural Information Processing Systems 22, pages 90–98. 2009. 62

[Bethge & Berens 2008] M. Bethge and P. Berens. *Near-Maximum Entropy Models for Binary Neural Representations of Natural Images.* In Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, volume 20, pages 97–104, Cambridge, MA, 2008. MIT Press. 94

[Bialek *et al.* 1991] W. Bialek, F. Rieke, RR de Ruyter van Steveninck and D. Warland. *Reading a neural code.* Science, vol. 252, no. 5014, pages 1854–1857, 1991. 5, 61, 63, 78

[Bishop *et al.* 2006] C.M. Bishop*et al.* Pattern recognition and machine learning. Springer New York, 2006. 72

[Boogaard 1986] H. Boogaard. *Maximum likelihood estimations in a nonlinear self-exciting point process model.* Biological Cybernetics, vol. 55, no. 4, pages 219–225, July 1986. 6

[Borisyuk *et al.* 1985] GN Borisyuk, RM Borisyuk, AB Kirillov, EI Kovalenko and VI Kryukov. *A new statistical method for identifying interconnections between neuronal network elements.* Biological Cybernetics, vol. 52, no. 5, pages 301–306, 1985. 6, 21

[Brillinger 1988] D.R. Brillinger. *Maximum likelihood analysis of spike trains of interacting nerve cells.* Biological Cybernetics, vol. 59, no. 3, pages 189–200, 1988. 6, 10, 14, 20, 21, 22

[Broderick *et al.* 2007] Tamara Broderick, Miroslav Dudik, Gasper Tkacik, Robert E Schapire and William Bialek. *Faster solutions of the inverse pairwise Ising problem.* arXiv, vol. q-bio.QM, page 0712.2437, Dec 2007. 97

[Brown *et al.* 2002] E.N. Brown, R. Barbieri, V. Ventura, R.E. Kass and L.M. Frank. *The time-rescaling theorem and its application to neural spike train data analysis.* Neural Computation, vol. 14, no. 2, pages 325–346, 2002. 6

[Brown *et al.* 2003] E.N. Brown, R. Barbieri, U.T. Eden and L.M. Frank. Likelihood methods for neural spike train data analysis, chapitre 9, pages 253–286. CRC Press, Boca Raton, 2003. 6

[Brown *et al.* 2004] E.N. Brown, R.E. Kass and P.P. Mitra. *Multiple neural spike train data analysis: state-of-the-art and future challenges.* Nature Neuroscience, vol. 7, no. 5, pages 456–461, 2004. 6

[Burkitt 2006] A.N. Burkitt. *A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input.* Biological cybernetics, vol. 95, no. 1, pages 1–19, 2006. 11

[Buzsaki 2004] G. Buzsaki. *Large-scale recording of neuronal ensembles.* Nature Neuroscience, vol. 7, no. 5, pages 446–451, 2004. 93

[Carr & Konishi 1990] C. E. Carr and M. Konishi. *A circuit for detection of interaural time differences in the brain stem of the barn owl.* J Neurosci, vol. 10, no. 10, pages 3227–3246, Oct 1990. 38

[Chib 1995] S. Chib. *Marginal Likelihood from the Gibbs Output.* Journal of the American Statistical Association, vol. 90, no. 432, 1995. 32

[Chichilnisky 2001] EJ Chichilnisky. *A simple white noise analysis of neuronal light responses.* Network: Computation in Neural Systems, vol. 12, no. 2, pages 199–213, 2001. 20

[Chornoboy *et al.* 1988] E.S. Chornoboy, L.P. Schramm and A.F. Karr. *Maximum likelihood identification of neural point process systems.* Biological Cybernetics, vol. 59, no. 4, pages 265–275, 1988. 6, 20, 21, 22

[Coleman & Sarma 2010] T.P. Coleman and S.S. Sarma. *A Computationally Efficient Method for Nonparametric Modeling of Neural Spiking Activity with Point Processes.* Neural Computation, pages 1–29, 2010. 6

[Cox & Isham 1980] D.R. Cox and V. Isham. Point processes. Chapman & Hall/CRC, 1980. 5

[Cox & Wermuth 1999] D. R. Cox and Nanny Wermuth. *Likelihood Factorizations for Mixed Discrete and Continuous Variables.* Scandinavian Journal of Statistics, vol. 26, no. 2, pages 209–220, June 1999. 99

[Cox 1972] DR Cox. *The statistical analysis of dependencies in point processes.* Stochastic Point Processes: Statistical Analysis, Theory, and Applications, pages 55–66, 1972. 6

[Cunningham *et al.* 2008] J.P. Cunningham, K.V. Shenoy and M. Sahani. *Fast Gaussian process methods for point process intensity estimation.* Proceedings

of the 25th international conference on Machine learning, pages 192–199, 2008. 83, 90, 91

[Daley & Vere-Jones 2005] D.J. Daley and D. Vere-Jones. An introduction to the theory of point processes, volume i: Elementary theory and methods. Springer, 2005. 5

[Daley & Vere-Jones 2008] DJ Daley and D. Vere-Jones. An Introduction to the Theory of Point Processes. Vol. II: General Theory and Structure. Springer. New York, 2008. 5, 25

[Dayan *et al.* 2001] P. Dayan, L.F. Abbott and L. Abbott. Theoretical neuroscience: Computational and mathematical modeling of neural systems. MIT Press, 2001. 19

[De Boer & Kuyper 1968] R. De Boer and P. Kuyper. *Triggered correlation.* IEEE transactions on bio-medical engineering, vol. 15, no. 3, page 169, 1968. 5, 19

[Donoho & Stodden 2006] D.L. Donoho and V. Stodden. *Breakdown point of model selection when the number of variables exceeds the number of observations.* In Proceedings of the International Joint Conference on Neural Networks, pages 16–21, 2006. 29

[Fairhall *et al.* 2006] A.L. Fairhall, C.A. Burlingame, R. Narasimhan, R.A. Harris, J.L. Puchalla and M.J. Berry. *Selectivity for multiple stimulus features in retinal ganglion cells.* Journal of neurophysiology, vol. 96, no. 5, page 2724, 2006. 39

[Georgopoulos *et al.* 1982] A.P. Georgopoulos, J.F. Kalaska, R. Caminiti and J.T. Massey. *On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex.* Journal of Neuroscience, vol. 2, no. 11, page 1527, 1982. 61

[Gerstner & Kistler 2002] W. Gerstner and W.M. Kistler. Spiking neuron models: Single neurons, populations, plasticity. Cambridge University Press, 2002. 10, 63, 64

[Gerwinn *et al.* 2008] S. Gerwinn, J. Macke, M. Seeger and M. Bethge. *Bayesian Inference for Spiking Neuron Models with a Sparsity Prior.* In J. C. Platt, D. Koller, Y. Singer and S. Roweis, editeurs, Advances in Neural Information Processing Systems 20, pages 529 – 536. MIT Press, Cambridge, MA, 2008. 7

[Gerwinn *et al.* 2009a] S. Gerwinn, P. Berens and M. Bethge. *A joint maximum-entropy model for binary neural population patterns and continuous signals.* In Advances in Neural Information Processing Systems. 2009. 8

[Gerwinn *et al.* 2009b] Sebastian Gerwinn, Jakob H. Macke, Matthias Bethge and Gerwinn Sebastian Macke Jakob H Bethge Matthias. *Bayesian population decoding of spiking neurons.* Frontiers in Computational Neuroscience, 2009. 7

[Gerwinn *et al.* 2010] Sebastian Gerwinn, Jakob H. Macke and Matthias Bethge. *Bayesian inference for generalized linear models for spiking neurons.* Frontiers in Computational Neuroscience, vol. 4, 2010. 7

[Harris *et al.* 2003] K.D. Harris, J. Csicsvari, H. Hirase, G. Dragoi and G. Buzsáki. *Organization of cell assemblies in the hippocampus.* Nature, vol. 424, no. 6948, pages 552–556, 2003. 6, 21

[Heskes *et al.* 2002] T. Heskes, O. Zoeter, A. Darwiche and N. Friedman. *Expectation propagation for approximate inference in.* In Proceedings UAI-2002, pages 216–233, 2002. 34

[Hodgkin & Huxley 1952] A. L. Hodgkin and A. F. Huxley. *A quantitative description of membrane current and its application to conduction and excitation in nerve.* J Physiol, vol. 117, no. 4, pages 500–544, Aug 1952. 6

[Hubel & Wiesel 1962] DH Hubel and TN Wiesel. *Receptive fields, binocular interaction and functional architecture in the cat's visual cortex.* The Journal of Physiology, vol. 160, no. 1, page 106, 1962. 20

[Huys *et al.* 2007] Q.J.M. Huys, R.S. Zemel, R. Natarajan and P. Dayan. *Fast Population Coding.* Neural Computation, vol. 19, no. 2, pages 404–441, 2007. 62

[Ising 1925] E. Ising. *Beitrag zur theorie des ferromagnetismus.* Zeitschrift f "ur Physik A Hadrons and Nuclei, vol. 31, no. 1, pages 253–258, 1925. 17

[Jolivet *et al.* 2006] R. Jolivet, A. Rauch, H.R. Lüscher and W. Gerstner. *Predicting spike timing of neocortical pyramidal neurons by simple threshold models.* Journal of computational neuroscience, vol. 21, no. 1, pages 35–49, 2006. 12, 66, 80

[Jolivet *et al.* 2008] Renaud Jolivet, Felix Schürmann, Thomas Berger, Richard Naud, Wulfram Gerstner and Arnd Roth. *The quantitative single-neuron*

modeling competition. Biological Cybernetics, vol. 99, no. 4, pages 417–426, November 2008. 6

[Julier & Uhlmann 1997] S.J. Julier and J.K. Uhlmann. *A new extension of the Kalman filter to nonlinear systems.* In International Symposium on Aerospace/Defense Sensing, Simulation, and Controls, volume 3, 1997. 73

[Kerr & Denk 2008] J. N. D. Kerr and W. Denk. *Imaging in vivo: watching the brain in action.* Nature Review Neuroscience, vol. 9, no. 3, page 195âĂŞ205, 2008. 5

[Koyama & Paninski 2009] S. Koyama and L. Paninski. *Efficient computation of the maximum a posteriori path and parameter estimation in integrate-and-fire and more general state-space models.* Journal of Computational Neuroscience, pages 1–17, 2009. 14, 34, 51

[Krzanowski 1993] W. Krzanowski. *The location model for mixtures of categorical and continuous variables.* Journal of Classification, vol. 10, no. 1, pages 25–49, 1993. 94, 98, 99, 105

[Kulkarni & Paninski 2007] J.E. Kulkarni and L. Paninski. *Common-input models for multiple neural spike-train data.* Network: Computation in Neural Systems, vol. 18, no. 4, pages 375–407, 2007. 25

[Kuss & Rasmussen 2005] M. Kuss and C.E. Rasmussen. *Assessing approximate inference for binary Gaussian process classification.* The Journal of Machine Learning Research, vol. 6, page 1704, 2005. 34

[Lauritzen & Wermuth 1989] S. L. Lauritzen and N. Wermuth. *Graphical Models for Associations between Variables, some of which are Qualitative and some Quantitative.* The Annals of Statistics, vol. 17, no. 1, pages 31–57, March 1989. 94, 98

[Lehmann & Casella 1998] E.L. Lehmann and G. Casella. Theory of point estimation. Springer Verlag, 1998. 58

[Lewi *et al.* 2008] J. Lewi, R. Butera and L. Paninski. *Sequential optimal design of neurophysiology experiments.* Neural Computation, vol. in press, 2008. 32, 34, 62, 91

[Lewicki & Olshausen 1999] M.S. Lewicki and B.A. Olshausen. *Probabilistic framework for the adaptation and comparison of image codes.* Journal of the Optical Society of America A, vol. 16, no. 7, pages 1587–1601, 1999. 20, 27

[Ma *et al.* 2006] W.J. Ma, J.M. Beck, P.E. Latham and A. Pouget. *Bayesian inference with probabilistic population codes.* Nature Neuroscience, vol. 9, pages 1432–1438, 2006. 9, 62

[MacKay 2003] D.J.C. MacKay. Information Theory, Inference and Learning Algorithms. Cambridge University Press, 2003. 34, 63, 83, 97

[Macke *et al.* 2008] Jakob Macke, Guenther Zeck and Matthias Bethge. *Receptive Fields without Spike-Triggering.* In J.C. Platt, D. Koller, Y. Singer and S. Roweis, editeurs, Advances in Neural Information Processing Systems 20, pages 969–976. MIT Press, Cambridge, MA, 2008. 20

[Macke *et al.* 2009] J.H. Macke, P. Berens, A.S. Ecker, A.S. Tolias and M. Bethge. *Generating Spike Trains with Specified Correlation Coefficients.* Neural Computation, vol. 21, no. 2, pages 1–27, 2009. 94

[Marmarelis & Naka 1972] P.Z. Marmarelis and K.I. Naka. *White-noise analysis of a neuron chain: An application of the Wiener theory.* Science, vol. 175, no. 4027, page 1276, 1972. 5, 19

[McCullagh & Nelder 1989] P. McCullagh and J. A. Nelder. Generalized linear models. Chapman & Hall/CRC, 1989. 20, 22

[Mineault *et al.* 2009] PJ Mineault, S. Barthelmé and CC Pack. *Improved classification images with sparse priors in a smooth basis.* Journal of Vision, vol. 9, pages 10–17, 2009. 20, 27

[Minka 2001] T.P. Minka. *A family of algorithms for approximate Bayesian inference.* PhD thesis, Massachusetts Institute of Technology, 2001. 21, 32, 73, 111

[Montemurro *et al.* 2008] M.A. Montemurro, M.J. Rasch, Y. Murayama, N.K. Logothetis and S. Panzeri. *Phase-of-firing coding of natural visual stimuli in primary visual cortex.* Current Biology, vol. 18, no. 5, pages 375–380, 2008. 62, 104

[Natarajan *et al.* 2008] R. Natarajan, Q.J.M. Huys, P. Dayan and R.S. Zemel. *Encoding and Decoding Spikes for Dynamic Stimuli.* Neural Computation, vol. 20, no. 9, pages 2325–2360, 2008. 62

[Neal 1996] R.M. Neal. Bayesian Learning for Neural Networks. Springer, 1996. 7

[Ng 2004] A.Y. Ng. *Feature selection, L 1 vs. L 2 regularization, and rotational invariance.* In Proceedings of the twenty-first international conference on Machine learning. ACM New York, NY, USA, 2004. 20, 27, 35

[Nickisch & Rasmussen 2008] H. Nickisch and C.E. Rasmussen. *Approximations for binary gaussian process classification.* Journal of Machine Learning Research, vol. 9, pages 2035–2078, 2008. 32

[Nykamp 2008] Duane Q Nykamp. *Pinpointing connectivity despite hidden nodes within stimulus-driven networks.* Phys Rev E Stat Nonlin Soft Matter Phys, vol. 78, no. 2 Pt 1, page 021902, Aug 2008. 25

[Okatan *et al.* 2005] M. Okatan, M.A. Wilson and E.N. Brown. *Analyzing Functional Connectivity Using a Network Likelihood Model of Ensemble Neural Spiking Activity.* Neural Computation, vol. 17, no. 9, pages 1927–1961, 2005. 6, 10, 14, 22

[Oksendal & Karsten 1998] B. Oksendal and B. Karsten. Stochastic Differential Equations: An Introduction with Applications. Springer, 1998. 11

[Olkin & Tate 1961] I. Olkin and R. F. Tate. *Multivariate Correlation Models with Mixed Discrete and Continuous Variables.* The Annals of Mathematical Statistics, vol. 32, no. 2, pages 448–465, June 1961. 94, 98

[Olshausen & Field 1996] B.A. Olshausen and D.J. Field. *Emergence of simple-cell receptive field properties by learning a sparse code for natural images.* Nature, vol. 381, no. 6583, pages 607–609, 1996. 90

[Opper & Winther 2000] M. Opper and O. Winther. *Gaussian Processes for Classification: Mean-Field Algorithms.* Neural Computation, vol. 12, no. 11, pages 2655–2684, 2000. 32

[Opper & Winther 2005] M. Opper and O. Winther. *Expectation consistent approximate inference.* The Journal of Machine Learning Research, vol. 6, pages 2177–2204, 2005. 32, 33, 111

[Paninski *et al.* 2004] Liam Paninski, Jonathan W Pillow and Eero P Simoncelli. *Maximum likelihood estimation of a stochastic integrate-and-fire neural encoding model.* Neural Computation, vol. 16, no. 12, pages 2533–2561, 2004. 6, 11, 12, 29, 66, 82, 90, 91

[Paninski *et al.* 2007] L. Paninski, J. Pillow and J. Lewi. *Statistical models for neural encoding, decoding, and optimal stimulus design.* Progress in brain research, vol. 165, page 493, 2007. 62, 63, 66, 67, 80, 83, 90, 91

[Paninski *et al.* 2008] Liam Paninski, Adrian Haith and Gabor Szirtes. *Integral equation methods for computing likelihoods and their derivatives in the stochastic integrate-and-fire model.* Journal of Computational Neuroscience, vol. 24, no. 1, pages 69–79, February 2008. 12, 51

[Paninski 2003] L. Paninski. *Convergence properties of three spike-triggered analysis techniques.* Network: Computation in Neural Systems, vol. 14, no. 3, pages 437–464, 2003. 6

[Paninski 2004] L. Paninski. *Maximum likelihood estimation of cascade point-process neural encoding models.* Network, vol. 15, no. 4, pages 243–262, 2004. 6, 10, 14, 20, 22, 23, 32

[Paradiso 1988] MA Paradiso. *A theory for the use of visual orientation information which exploits the columnar structure of striate cortex.* Biological Cybernetics, vol. 58, no. 1, pages 35–49, 1988. 62

[Pawitan 2001] Y. Pawitan. In all likelihood: statistical modelling and inference using likelihood. Oxford University Press, USA, 2001. 6

[Penrose 1955] R. Penrose. *A generalized inverse for matrices.* In Proceedings of the Cambridge Philosophical Society, volume 51, pages 406–413, 1955. 70

[Piessens *et al.* 1983] R. Piessens, E. de Doncker-Kapenga, CW Uberhuber and DK Kahaner. QUADPACK: A subroutine package for automatic integration. Springer Berlin, 1983. 34

[Pillow & Simoncelli 2002] J. Pillow and E.P. Simoncelli. *Biases in white noise analysis due to non-Poisson spike generation.* vol. 21, page 25, 2002. 65, 69

[Pillow & Simoncelli 2006] J. Pillow and E.P. Simoncelli. *Dimensionality reduction in neural models: an information-theoretic generalization of spike-triggered average and covariance analysis.* Journal of Vision, vol. 6, no. 4, pages 414–428, 2006. 25, 94, 99

[Pillow *et al.* 2005] J.W. Pillow, L. Paninski, V.J. Uzzell, E.P. Simoncelli and EJ Chichilnisky. *Prediction and Decoding of Retinal Ganglion Cell Responses with a Probabilistic Spiking Model.* Journal of Neuroscience, vol. 25, no. 47, pages 11003–11013, 2005. 21, 26, 40

[Pillow *et al.* 2008] J. Pillow, J. Shlens, L. Paninski, A. Sher, A.M. Litke, EJ Chichilnisky and E.P. Simoncelli. *Spatio-temporal correlations and visual signalling in a complete neuronal population.* Nature, vol. 454, pages 995–999, 2008. 6, 21, 31, 40

[Pillow 2009] J. Pillow. *Time-rescaling methods for the estimation and assessment of non-Poisson neural encoding models.* In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams and A. Culotta, editeurs, Advances in Neural Information Processing Systems 22, pages 1473–1481. 2009. 6, 25

[Plesser & Gerstner 2000] H.E. Plesser and W. Gerstner. *Noise in Integrate-and-Fire Neurons: From Stochastic Input to Escape Rates.* Neural Computation, vol. 12, no. 2, pages 367–384, 2000. 51

[Pouget *et al.* 2000] A. Pouget, P. Dayan and R. Zemel. *Information processing with population codes.* Nature Reviews Neuroscience, vol. 1, no. 2, pages 125–132, 2000. 62

[Qi *et al.* 2004] Y.A. Qi, T.P. Minka, R.W. Picard and Z. Ghahramani. *Predictive automatic relevance determination by expectation propagation.* In Proceedings of the twenty-first International Conference on Machine Learning. ACM New York, NY, USA, 2004. 34

[Rao *et al.* 2002] R.P.N. Rao, B.A. Olshausen and M.S. Lewicki, editeurs. Probabilistic models of the brain. MIT Press, 2002. 62

[Rao 2005] R.P.N. Rao. *Hierarchical Bayesian inference in networks of spiking neurons.* Advances in Neural Information Processing Systems, vol. 17, pages 1113–20, 2005. 62

[Rasch *et al.* 2008] M.J. Rasch, A. Gretton, Y. Murayama, W. Maass and N.K. Logothetis. *Inferring spike trains from local field potentials.* Journal of Neurophysiology, vol. 99, no. 3, page 1461, 2008. 62

[Rasmussen & Williams 2006] C.E. Rasmussen and C.K.I. Williams. Gaussian Processes for Machine Learning. MIT Press, Cambridge, MA, USA, 2006. 34, 63, 83

[Rieke *et al.* 1997] F. Rieke, D. Warland, R.R. van Steveninck and W. Bialek. Spikes: exploring the neural code. MIT Press, Cambridge, MA, 1997. 25, 78, 80

[Risken 1989] H. Risken. The Fokker-Planck equation: Methods of solution and applications. Springer, 1989. 51

[Roudi *et al.* 2009a] Y. Roudi, E. Aurell and J.A. Hertz. *Statistical physics of pairwise probability models.* Frontiers in Computational Neuroscience, 2009. 94

[Roudi *et al.* 2009b] Y. Roudi, S. Nirenberg and P. Latham. *Pairwise Maximum Entropy Models for Studying Large Biological Systems: When They Can Work and When They Can't.* PLoS Comput Biol, vol. 5, no. 5, 2009. 94

[Roudi *et al.* 2009c] Y. Roudi, J. Tyrcha and J. Hertz. *The Ising Model for Neural Data: Model Quality and Approximate Methods for Extracting Functional Connectivity.* Phys. Rev. E, vol. 79, page 051915, February 2009. 10

[Rust *et al.* 2005] N.C. Rust, O. Schwartz, J.A. Movshon and E.P. Simoncelli. *Spatiotemporal elements of macaque v1 receptive fields.* Neuron, vol. 46, no. 6, pages 945–956, 2005. 25, 35, 49, 51

[Schneidman *et al.* 2006] E. Schneidman, M. J. Berry, R. Segev and W. Bialek. *Weak pairwise correlations imply strongly correlated network states in a neural population.* Nature, vol. 440, no. 7087, pages 1007–1012, April 2006. 10, 94, 102, 104

[Schwartz *et al.* 2002] O. Schwartz, EJ Chichilnisky and E.P. Simoncelli. *Characterizing neural gain control using spike-triggered covariance.* In Advances in Neural Information Processing Systems 14: Proceedings of the 2002 [sic] Conference, page 269. MIT Press, 2002. 20, 94, 99

[Seeger *et al.* 2007] M. Seeger, S. Gerwinn and M. Bethge. *Bayesian inference for sparse generalized linear models.* Lecture Notes in Computer Science, vol. 4701, page 298, 2007. 7

[Seeger 2005] M. Seeger. *Expectation propagation for exponential families.* Rapport technique, University of California at Berkeley, 2005. 32, 111, 115

[Seeger 2008] M. Seeger. *Bayesian inference and optimal design for the sparse linear model.* The Journal of Machine Learning Research, vol. 9, pages 759–813, 2008. 32, 113

[Seung & Sompolinsky 1993] HS Seung and H. Sompolinsky. *Simple models for reading neuronal population codes.* Proceedings of the National Academy of Sciences, vol. 90, no. 22, page 10749, 1993. 62

[Seydnejad & Kitney 2001] SR Seydnejad and RI Kitney. *Time-varying threshold integral pulse frequency modulation.* IEEE Transactions on Biomedical Engineering, vol. 48, no. 9, pages 949–962, 2001. 62, 69

[Shadlen & Newsome 1998] M.N. Shadlen and W.T. Newsome. *Noise, neural codes and cortical organization.* Find Curr Opin Cognit Neurosci, vol. 4, pages 569–79, 1998. 4, 9

[Shadlen *et al.* 1996] MN Shadlen, KH Britten, WT Newsome and JA Movshon. *A computational analysis of the relationship between neuronal and behavioral responses to visual motion.* Journal of Neuroscience, vol. 16, no. 4, page 1486, 1996. 62

[Sharpee *et al.* 2004] T. Sharpee, N.C. Rust and W. Bialek. *Analyzing neural responses to natural signals: maximally informative dimensions.* Neural Computation, vol. 16, no. 2, pages 223–250, 2004. 20

[Shlens *et al.* 2006] J. Shlens, G. D. Field, J. L. Gauthier, M. I. Grivich, D. Petrusca, A. Sher, A.M. Litke and E.J. Chichilnisky. *The Structure of Multi-Neuron Firing Patterns in Primate Retina.* Journal of Neuroscience, vol. 26, no. 32, pages 8254–8266, August 2006. 94, 102

[Shlens *et al.* 2008] Jonathon Shlens, Fred Rieke and E. J. Chichilnisky. *Synchronized firing in the retina.* Current Opinion in Neurobiology, vol. 18, no. 4, pages 396–402, August 2008. 94

[Shlens *et al.* 2009] J. Shlens, G. D. Field, J. L. Gauthier, M. Greschner, A. Sher, A. M. Litke and E. J. Chichilnisky. *The Structure of Large-Scale Synchronized Firing in Primate Retina.* Journal of Neuroscience, vol. 29, no. 15, page 5022, 2009. 93, 94

[Simoncelli *et al.* 2004] E. Simoncelli, L. Paninski and J. Pillow. The cognitive neurosciences, chapitre 23, pages 327–338. MIT Press, Cambridge, MA, USA, 2004. 14, 20, 22, 49

[Stein 1967] RB Stein. *Some Models of Neuronal Variability.* Biophysical Journal, vol. 7, no. 1, page 37, 1967. 62

[Steinke *et al.* 2007] F. Steinke, M. Seeger and K. Tsuda. *Experimental design for efficient identification of gene regulatory networks using sparse Bayesian models.* BMC Systems Biology, vol. 1, no. 1, page 51, 2007. 20, 27

[Stevenson *et al.* 2008] I.H. Stevenson, J.M. Rebesco, L.E. Miller and K.P. Körding. *Inferring functional connections between neurons.* Current Opinion in Neurobiology, vol. 18, pages 582–588, 2008. 22

[Tang *et al.* 2008] A. Tang, D. Jackson, J. Hobbs, W. Chen, J.L. Smith, H. Patel, A. Prieto, D. Petrusca, M.I. Grivich, A. Sher*et al. A Maximum Entropy Model Applied to Spatial and Temporal Correlations from Cortical Networks In Vitro.* Journal of Neuroscience, vol. 28, no. 2, pages 505–518, 2008. 10, 94

[Tibshirani 1996] R. Tibshirani. *Regression shrinkage and selection via the lasso.* Journal of the Royal Statistical Society. Series B (Methodological), vol. 58, no. 1, pages 267–288, 1996. 27, 29, 34

[Tolias *et al.* 2007] A. S. Tolias, A. S. Ecker, A. G. Siapas, A. Hoenselaar, G. A. Keliris and N. K. Logothetis. *Recording chronically from the same neurons in awake, behaving primates.* Journal of Neurophysiology, vol. 98, no. 6, pages 3780–3790, 2007. 5

[Touryan *et al.* 2002] J. Touryan, B. Lau and Y. Dan. *Isolation of Relevant Visual Features from Random Stimuli for Cortical Complex Cells.* Journal of Neuroscience, vol. 22, no. 24, page 10811, 2002. 49

[Trong & Rieke 2008] Philipp Khuc Trong and Fred Rieke. *Origin of correlated activity between parasol retinal ganglion cells.* Nat Neurosci, vol. 11, no. 11, pages 1343–1351, Nov 2008. 57

[Truccolo & Donoghue 2007] Wilson Truccolo and John P. Donoghue. *Nonparametric Modeling of Neural Point Processes via Stochastic Gradient Boosting Regression.* Neural Computation, vol. 19, no. 3, pages 672–705, March 2007. 6

[Truccolo *et al.* 2005] W. Truccolo, U.T. Eden, M.R. Fellows, J.P. Donoghue and E.N. Brown. *A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects.* Journal of Neurophysiology, vol. 93, no. 2, pages 1074–1089, 2005. 6, 22

[Truccolo *et al.* 2009] W. Truccolo, L.R. Hochberg and J.P. Donoghue. *Collective Dynamics in Human and Monkey Sensorimotor Cortex: Predicting Single Neuron Spikes.* Nature Neuroscience, 2009. 21

[Tuckwell 1988] H.C. Tuckwell. Introduction to Theoretical Neurobiology. Cambridge University Press, 1988. 10, 62, 64

[Van Steveninck & Bialek 1988] R.D.R. Van Steveninck and W. Bialek. *Real-time performance of a movement-sensitive neuron in the blowfly visual system: coding and information transfer in short spike sequences.* Proceedings of the Royal Society of London. Series B, Biological Sciences, vol. 234, no. 1277, pages 379–414, 1988. 20, 25, 49

[Victor & Purpura 1997] J.D. Victor and K.P. Purpura. *Metric-space analysis of spike trains: theory, algorithms and application.* Network: computation in neural systems, vol. 8, no. 2, pages 127–164, 1997. 57, 95, 103

[Weisberg & Welsh 1994] S. Weisberg and AH Welsh. *Adapting for the missing link.* The Annals of Statistics, vol. 22, no. 4, pages 1674–1700, 1994. 6

[Wiener & Richmond 2003] M.C. Wiener and B.J. Richmond. *Decoding Spike Trains Instant by Instant Using Order Statistics and the Mixture-of-Poissons Model.* Journal of Neuroscience, vol. 23, no. 6, page 2394, 2003. 63

[Wightman & Kistler 1992] F. L. Wightman and D. J. Kistler. *The dominant role of low-frequency interaural time differences in sound localization.* J Acoust Soc Am, vol. 91, no. 3, pages 1648–1661, Mar 1992. 38

[Yu *et al.* 2008] Shan Yu, Debin Huang, Wolf Singer and Danko Nikolic. *A Small World of Neuronal Synchrony.* Cereb. Cortex, vol. 18(12), pages 2891–2901, April 2008. 94

[Zeck *et al.* 2005] Gunther M Zeck, Quan Xiao and Richard H Masland. *The spatial filtering properties of local edge detectors and brisk-sustained retinal ganglion cells.* Eur J Neurosci, vol. 22, no. 8, pages 2016–26, 2005. 5, 45