Low-rank graphical models and Bayesian inference in the statistical analysis of noisy neural data

Carl Smith

Submitted in partial fulfillment of the
requirements for the degree
of Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2013

ABSTRACT

Low-rank graphical models and Bayesian inference in the statistical analysis of noisy neural data

Carl Smith

We develop new methods of Bayesian inference, largely in the context of analysis of neuroscience data. The work is broken into several parts. In the first part, we introduce a novel class of joint probability distributions in which exact inference is tractable. Previously it has been difficult to find general constructions for models in which efficient exact inference is possible, outside of certain classical cases. We identify a class of such models that are tractable owing to a certain "low-rank" structure in the potentials that couple neighboring variables. In the second part we develop methods to quantify and measure information loss in analysis of neuronal spike train data due to two types of noise, making use of the ideas developed in the first part. Information about neuronal identity or temporal resolution may be lost during spike detection and sorting, or precision of spike times may be corrupted by various effects. We quantify the information lost due to these effects for the relatively simple but sufficiently broad class of Markovian model neurons. We find that decoders that model the probability distribution of spike-neuron assignments significantly outperform decoders that use only the most likely spike assignments. We also apply the ideas of the low-rank models from the first section to defining a class of prior distributions over the space of stimuli (or other covariate) which, by conjugacy, preserve the tractability of inference. In the third part, we treat Bayesian methods for the estimation of sparse signals, with application to the locating of synapses in a dendritic tree. We develop a compartmentalized model of the dendritic tree. Building on previous work that applied and generalized ideas of least angle regression to obtain a fast Bayesian solution to the resulting estimation problem, we describe two other approaches to the same problem, one employing a horseshoe prior and the other using various spike-and-slab priors. In the last part, we revisit the low-rank mod-

els of the first section and apply them to the problem of inferring orientation selectivity maps from noisy observations of orientation preference. The relevant low-rank model exploits the self-conjugacy of the von Mises distribution on the circle. Because the orientation map model is loopy, we cannot do exact inference on the low-rank model by the forward backward algorithm, but block-wise Gibbs sampling by the forward backward algorithm speeds mixing. We explore another von Mises coupling potential Gibbs sampler that proves to effectively smooth noisily observed orientation maps.

Table of Contents

1 Introduction **1**

2 Low rank models: Tractable inference in continuous, non-Gaussian graphical models **8**

2.1 Introduction 8

2.2 Related work 10

2.3 Low-rank Markov chains 11

2.4 Examples 15

2.4.1 Beta-binomial and Dirichlet-multinomial time series 15

2.4.2 Smoothing conjugate priors for multinomial data 17

2.4.3 Phase data 18

2.5 Experiments 19

2.6 Discussion 20

2.7 Appendix 21

3 Information loss in spike trains due to spike time jitter and ambiguity of neuronal identity **30**

3.1 Abstract 30

3.2 Introduction 31

3.3 Sources of information loss 33

3.3.1 Spike time jitter 33

3.3.2 Spike identity loss 34

3.4 Model assumptions 35

3.4.1 Spike time jitter . . . 36
3.4.2 Identity loss . . . 36
3.5 Stimulus decoding . . . 36
3.5.1 Gibbs sampling and Rao-Blackwellizaition . . . 37
3.5.2 The stimulus decoder Gibbs sampler . . . 38
3.5.3 Hidden Markov models . . . 39
3.5.4 Sampling from the posterior state distribution $p(Q|X,Y)$ . . . 41
3.5.5 Sampling from the posterior stimulus distribution $p(X|Q,Y)$ . . . 47
3.6 Results . . . 50
3.6.1 Spike time jitter . . . 50
3.6.2 Identity loss . . . 53
3.7 Conclusions and extensions . . . 56

4 Bayesian inference of sparse synaptic connectivity with spike-and-slab models 60

4.1 Abstract . . . 60
4.2 Introduction . . . 61
4.3 Dynamical model . . . 63
4.4 Bayesian approaches . . . 65
4.4.1 The horseshoe model . . . 65
4.4.2 Spike-and-slab models . . . 69
4.5 Results . . . 80
4.6 Conclusions . . . 81

5 Inferring orientation selectivity maps with low-rank von Mises and related priors 86

5.1 Abstract . . . 86
5.2 Introduction . . . 87
5.3 Low-rank models revisited . . . 87
5.4 The von Mises distribution and smoothing potential . . . 90
5.4.1 The von Mises distribution . . . 90

5.4.2 The von Mises smoothing prior 91
5.5 Loopy orientation map model 91
5.5.1 Smooth prior and posterior distributions of an orientation map . . . 92
5.5.2 The conditional distribution of a column 94
5.5.3 Sampling a column block-wise by the filter forward sample backward algorithm 95
5.5.4 Computing marginal distributions by the forward backward algorithm 97
5.5.5 Low-rank prior draw 98
5.6 A simpler phase selectivity map model 99
5.7 Experimental data 99
5.8 Phase map reconstruction: simpler model 100
5.9 Phase map reconstruction: low-rank model 101

Bibliography **102**

List of Figures

2.1 Graphical models of latent variables coupled by auxiliary variables 10
(a) Graphical model with "smoothing" dependency between latent variables. 10
(b) Graphical model with latent "smoothing" dependency induced by auxiliary variables. 10
(c) Factor graph corresponding to the graphical model in (b). 10
2.2 Maximum marginal likelihood selection of rank parameter 14
2.3 The inferred spiking probability density from spike train data 16
2.4 The Gibbs sampler solution compared to the exact solution. 28
2.5 The inferred probability density of angle in motion capture data. 29

3.1 Problem schematic. 34
3.2 A simple three-state neuron model. 37
3.3 An example sequence of samples from $p(X|\mathbf{Q})$ and $p(\mathbf{Q}|X, \mathbf{Y})$. 40
3.4 The inferred spiking probability density from spike train data. 50
3.5 Effects of jitter scale on decoding accuracy for stimulus drawn from the beta-binomial prior. 51
3.6 Accuracy of reconstruction in the jitter case. 53
3.7 Reconstructions of different stimuli both for no jitter and for jitter using the independent uniform prior. 54
3.8 Reconstructions of different stimuli both for no jitter and for jitter using a low-rank prior. 55
3.9 Sample reconstructions of a simple sawtooth stimulus for three examples of monotonically decreasing spike sorting difficulty. 57

3.10 Reconstruction error for three spike-sorting decoders. 59

4.1 Schematic of method. 62
4.2 Estimates of synaptic weights for a small neuron with true nonzero weights drawn from a log-normal distribution. The noise parameter here is $C_y = 0.01$. 82
4.3 Estimates of synaptic weights with noise parameter $C_y = 0.1$. 83
4.4 Estimates of synaptic weights with noise parameter $C_y = 1$. 84
4.5 Estimates of synaptic weights with noise parameter $C_y = 10$. 85

5.1 Low-rank model construction. 88
5.2 The von Mises smoother transition kernel. 92
5.3 A graphical model of orientation selectivity, omitting observed nodes and their dependencies. 93
5.4 A graphical model of orientation selectivity, including observed nodes and their dependencies. 93
5.5 A graphical model of orientation selectivity, including observed nodes and auxiliary variables and their dependencies. 96
5.6 A sample grid of angles drawn from the von Mises orientation map prior. . 98
5.7 Selectivity distributions and neuron-neuron potential for the simple model . 103
5.8 Progress of the Gibbs sampler for the simpler phase selectivity model applied to data from spinal neuron recordings. 105
5.9 Maps of the errors in the estimate at different locations against that of the subsampled observation. 107
5.10 Progress of the Gibbs sampler for the simpler phase selectivity model applied to different data from spinal neuron recordings. 109
5.11 Selectivity distributions and neuron-neuron potential for the low-rank model 111
5.12 Progress of the Gibbs sampler for the low-rank phase selectivity model applied to the same data from spinal neuron recordings. 113
5.13 Progress of the Gibbs sampler for the low-rank phase selectivity model applied to different data from spinal neuron recordings. 115

Acknowledgments

If I have grown as a scientist and thinker over the past five years – and I do hope that I have – then I owe it primarily to interactions with my colleagues and mentors at Columbia, in the Statistics department, at the Center for Theoretical Neuroscience, and in my home department of Chemistry. What I know for certain is that I could not have half completed the work in this dissertation without the help of Ari Pakman, David Pfau, Alex Ramirez, Kamiar Rahnama Rad, Yashar Ahmadian, Kolia Sadeghi, Eftychios Pnevmatikakis, Tim Machado, Greg Wayne, Avi Ziskind, Jessica Forde, Kui Tang, Jonathan Huggins, and many others, including those mentioned below.

My beginnings in the Chemistry department were formative and I must acknowledge the substantial influence on my way of approaching problems by Joel Eaves, Peter Mayer, and Tim Berkelbach. What I only realized later that I was taking for granted were the excited conversations about various mathematical problems and ideas had, often over bagels or coffee, with Brenda Rubenstein, Glen Hocky, Florian Elste, Emanuel Gull, Richard Darst, Jonathan Kummerfeld, and sporadically others. Thanks are due to Heidi Perry for helping to orient me to the Reichman group, and to Wolfgang Lechner for helping to orient me to the city, in the very beginning.

My advisor in the Chemistry department, David Reichman, welcomed me to the group with remarkable friendliness and with challenging readings and problems that quickly showed me the limits of my abilities, which he subsequently helped show me how to push. Long after I turned to problems in statistical neuroscience, he was no less accessible or willing to help me along. I am grateful for his friendship and guidance.

My experience in the Statistics department would have been far too normal without the presence of Frank Wood, whom I thank for letting me know when I was wrong or when my work was boring, and for helping me to make it correct and interesting.

I owe many thanks to Liam Paninski for his patience with a transplant from the Chemistry department with a lot of catching up to do, but more than that for all that I learned from him about statistics, neuroscience, and how to think. I cannot say that I learned how anyone is capable of being so productive, especially while appearing always so calm, but I did learn that it sometimes involves a few lengthy excited email threads at three in the morning, and that much can be learned from the simplest case.

Last but not least, I owe a great debt of gratitude to my thesis committee, which has included David Reichman and Liam Paninski but also Ken Miller, Jonathan Owen, and Louis Brus. Thank you to Ken Miller for asking hard questions that were the right questions. Thank you to Jonathan Owen and Louis Brus for your feedback on drafts of the thesis itself. Thank you all for your valuable time.

To my parents.

Chapter 1

Introduction

Much of this work concerns the application of Bayesian inference to problems in the analysis of data from neuroscience experiments. An explanation is appropriate, then, of the utility and appeal of Bayesian inference. Likewise, we must demonstrate the applicability of Bayesian inference to some problems in neuroscience.

Inference is the act of drawing conclusions from premises by the application of logic. In statistical inference we are typically interested in some system or process that generates data with an element of randomness, i.e. according to a probability distribution. Given some data (the premises) from this system, we want to draw some conclusions (estimates) about the data-generating distribution.

We often assume that the unknown distribution belongs to a parameterized set of distributions – a statistical model. Maximum likelihood estimation offers a principle for computing a consistent and efficient estimator[1] of the parameters, and so of the distribution: for the given data, choose the value of the parameter that maximizes the data-generating distribution, which we call the likelihood.

Bayesian inference takes us further. If, before collecting any data, we already have some estimate for the parameters, in the form of a so-called prior distribution, then by Bayes rule

[1]Roughly speaking, an estimator is a rule for computing an estimate given some data, a consistent estimator is one whose estimate converges to the true value as the amount of data increases, and an efficient estimator is one whose estimate approaches the true value the most quickly of any possible estimator as the amount of data increases.

we may extend maximum likelihood to form a new objective function – called the posterior – which is proportional to the product of the likelihood and the prior. The value of the parameter that maximizes the posterior is an efficient estimate that incorporates *a priori* information. This is called maximum *a posteriori* (MAP) estimation. What is more, beyond the maximizing point estimate, and depending on how we go about computing the MAP estimate, we may be able to compute the full posterior distribution, which fully represents our uncertainty in that estimate.

When we estimate by maximum likelihood, we can sometimes use the likelihood as a representation of our uncertainty. After all, the likeihood is a function of the parameter we wish to estimate. However, it is not necessarily a proper probability distribution over the parameter, because the likelihood is defined as a distribution over the data, given a fixed parameter. This can lead to problems both of computation and interpretation. In the Bayesian context, the prior distribution ensures that the posterior is normalizable, i.e. that it is a proper probability distribution.

Information about uncertainty can be exceedingly helpful in deciding how to act based on the output of the inference algorithm, and can recommend one inference method over another when both are known to be about as accurate in general but each is better suited to certain problem regimes. When fully Bayesian methods are passed over in favor of other methods, it is often due to computational problems (technically speaking, samplers that are very slow to mix, algorithms that scale poorly with problem size, and so on). Therefore, there is a need for improved models and algorithms that will make fully Bayesian inference more feasible. Much of the following work is in response to that need.

Inference problems arise in many neuroscience contexts. For example, brain-machine interfaces (BMIs) aim to convey information – sensory information, motor intent, etc. – between the brain and the outside world. Implementations sometimes involve the embedding of an array of microelectrodes into the outermost layers of exposed neural tissue. These electrodes record extracellular voltage over time, and so too they indirectly record firing activity of nearby cells, i.e. spike trains. Ultimately, the device should transduce this signal and produce, say, some motor output, such as the movement of a mechanical limb, or of a cursor on a screen. The device must infer, from the recorded spike trains, the motor intent

of the subject. BMIs represent an important research problem with potentially dramatic clinical applications. They also provide a good example of the kind of technology that good and computationally tractable statistical inference algorithms can afford us.

In this work we apply new methods in Bayesian inference to four problems in neuroscience. First, we briefly treat the estimation of neuronal firing rate. Second, we develop inference methods for spike-train decoding in the presence of either of two known sources of noise: 1) jitter in the firing times and 2) ambiguity of neuronal identity in a population of neurons of mixed type. Third, we address the problem of mapping the dendritic arbor – finding where precisely synapses occur. Last, we treat the estimation of phase selectivity maps from noisy or subsampled data.

As we have already remarked, inference methods that are appealing in principle for desirable properties of their estimators are only useful if they are computationally feasible. As the size of the problem grows (as the length of the spike-train increases; as the dendritic arbor is further compartmentalized; as the selectivity map becomes larger), the computational cost of inference must grow reasonably slowly. In BMIs, discussed above, one can easily understand that it is crucial for the device to take its constant flow of input and compute decisions in real time, so that speeding up this decision process is critically important. In each of the applications in this work we have endeavored to make the cost of the inference method as low as is theoretically possible. To that end, we have developed a broad class of probability models for which inference requires exceptionally little computing time and storage.

These special models are exceptional in that they extend fast hidden Markov model (HMM) inference algorithms beyond the domain in which they are typically applied. These are continuous and non-Gaussian models that are amenable to standard tractable inference methods that are typically thought to only apply to discrete or linear-Gaussian models. (By a continuous model I mean a joint distribution over many continuous variables. Such distributions are frequently represented with nodes (variables) and edges (potentials coupling the variables) in a so-called graphical model.) The standard forward backward algorithm from the theory of HMMs (whose variables are discrete) may be used to perform inference on these special models whose variables may nonetheless be continuous. This is owing to a

special aspect of the structure of these models which we describe below.

We will briefly motivate the use of low-rank models and leave the details to the main text. A HMM consists of two sets of variables: hidden variables $\{q_t\}$ and observed variables $\{y_t\}$, each indexed by, for instance, discrete time intervals t. The set of hidden variables form a graphical model arranged as a straight chain. This means that there is a linear order to the variables (e.g., number of cars on a road at different discrete time intervals) and that adjacent variables (consecutive time intervals) are coupled by pairwise potentials (probability densities). The observed variables are coupled to the chain of hidden variables by other pairwise potentials. The graphical model the represents this joint distribution resembles a comb. There is a backbone of hidden variables connected in a chain, and attached to each hidden variable is an observed variable, connected by an edge.

The crucial assumptions inherent to such a model are 1) that the value of a hidden variable q_t at time t depends only on the value of the hidden variable q_{t-1} at time $t-1$, and 2) that the value of a given observed variable y_t at time t depends only on the value of the hidden variable q_t at time t. The first assumption means that if we want to estimate q_t, and we know the value of q_{t-1}, then we do not benefit at all by knowing additionally the values of any of the other variables. The second assumption similarly means that if we want to estimate y_t and we know q_t, then there is no point in measuring any of the other variables, observed or hidden.

The appeal of HMMs stems partly from the fast and scalable algorithms that exist for performing inference in them. Also, however, there are many systems one might like to study in which these so-called Markov assumptions are very plausible.

We will not go here into much more detail about the forward backward method for HMMs. What is important for our work is that the inference algorithm involves the computation of certain integrals, and that the more possible discrete values the hidden variables can take – the larger their state space – the more of those integrals we must compute. On the face of it, then, it may seem impossible to allow the hidden variables to be continuous. We would have to compute infinitely many integrals, a continuum in fact. What we discovered and what we report in the main text is that for certain choices of coupling potentials, the continuous variables are naturally integrated away during the recursion, leaving

a finite number of integrals to compute and rendering inference tractable. This is true for potentials that are separable functions, i.e. $\phi(q_t, q_{t+1}) = f(q_t)g(q_{t+1})$, but complete factorization is the condition of independence of the two variables, which would reduce the HMM to a set of independent variables, which is a less interesting problem. The key is that the same tractability holds for potentials that are sums of separable potentials, i.e. $\phi(q_t, q_{t+1}) = \sum_i f_i(q_t)g_i(q_{t+1})$. This opens up a wide space of models. Because a sum of separable potentials naturally recalls the decomposition of a matrix into a sum of products of rank-one matrices, and because most practical potentials of this sort are sums over relatively few terms, we call these "low-rank" models.

In the case of the problem of firing rate estimation, we make use of a low-rank model as a prior that promotes gradually varying firing rates. In the case of the spike train decoding application, we draw attention to a subclass of low-rank models that are conjugate to the likelihood in our model of spike train generation. Thus we provide a broad family of available prior distributions over the values of the time-varying input to the neuron (such as a stimulus). In the case of phase selectivity map estimation, we consider again a family of low-rank priors over the phase selectivity map. This illustrates that low-rank models exist even when the parameters being estimated live in structured state spaces, such as, here, the unit circle and, in the case of firing rate estimation, the postive reals.

In all of these examples, low-rank models are used to help introduce prior belief about the estimated parameter without compromising the tractability of inference. Even if maximum likelihood for a given model is tractable, introducing an arbitrary prior distribution will almost always render inference by maximum *a posteriori* less tractable. In each of the applications described above, it is the conjugacy of the prior distribution used to the likelihood of the generative model that ensures that inference remains feasible.

For the same reason, conjugacy plays an important role in inference in the application to locating synapses. In that case, we start with a model whose likelihood is Gaussian. Since we already believe that there will be few synapses – i.e. few nonzero elements to the vector of synaptic weights – we experiment with imposing several different prior distributions (over the synaptic weights vector) that promote sparsity of the estimate. In the case of so-called "spike-and-slab" priors, which are mixture distributions each consisting of a scaled

delta function plus a scaled continuous distribution, we are able to perform exact inference tractably when the slab component of the prior is conjugate to the model likelihood. When the slab and likelihood are not conjugate, then we must resort to approximate inference methods.

Our hope is that the theory and experiments described in the following will leave the reader convinced of two points: 1) that Bayesian inference is a powerful tool with wide applicability to the analysis of neuroscience data, and 2) that low-rank models render many applications of Bayesian inference – in neuroscience or elsewhere – tractable by extending HMM theory to a broad class of continuous models.

In the next chapter, we begin with a detailed look at low-rank models: their definition, their special structure, certain examples, and numerical experiments. These ideas are invoked later in Chapters 3 and 5 and comprise perhaps the most substantive novel contribution of the overall work. In Chapter 3, we focus on the problem of spike-train decoding in the presence of known sources of noise. We establish a broad class of generative models and a broad class of priors conjugate to them. We derive inference methods that retain tractability by the forward backward technique in spite of the presence of noise, and we present the results of several numerical experiments that demonstrate the performance of these methods. In Chapter 4, we turn our attention to the synapse locating setting. Here we pick up where previous related work left off, describing the generative model and previous approaches to sparsifying the estimated synaptic weights vector. We then present a few other approaches and compare their performance in numerical experiments. In Chapter 5, we discuss the problem of phase selectivity map estimation. We introduce a simple generative model and two families of conjugate prior distributions, one a low-rank model and the other a family of simpler distributions on whom only approximate inference is possible.

Notations are consistent within each chapter and largely across chapters, and are explained in the main text. A few common points of notation merit mention here. Unless otherwise stated, $p(x)$ is a probability density function over the random variable x. Probability mass functions over discrete random variables are considered special cases of density

functions and can be thought of as weighted sums of Dirac delta functions. The conditional distribution representing the probability the value of x given some value for y is written $p(x|y)$. We write $x \sim p(x)$ to denote that the random variable x is drawn from the distribution with density function $p(x)$, but we will also sometimes write the name of the distribution instead of its density function, as in $x \sim \text{Bernoulli}(x; p)$ meaning that x is drawn from the Bernoulli distribution with success rate p.

In probability and statistics, it is customary to write a capital letter to denote a random variable and the corresponding lowercase letter to denote the random variable's value. E.g., $X = x$ means that the random variable X takes the value x. Contrary to this tradition, in our work capital letters usually denote sets of random variables, such as time series, while lowercase letters denote particular random variables, e.g. $X = X_{1:T} = \{x_t\}_{t=1}^{T}$. Here x_t is the random variable at position (e.g. time) t. The subscript $i : j$ refers to elements i through j, inclusive, of the set of random variables of which it is the subscript. T is usually the length of a linear chain of random variables, even when they are not indexed by time. Other notation is defined in particular chapters.

Chapter 2

Low rank models: Tractable inference in continuous, non-Gaussian graphical models

Constructing tractable dependent probability distributions over structured continuous random vectors is a central problem in statistics and machine learning. It has proven difficult to find general constructions for models in which efficient exact inference is possible, outside of the classical cases of models with restricted graph structure (chain, tree, etc.) and linear-Gaussian or discrete potentials. In this work we identify a tree graphical model class in which exact inference can be performed efficiently, owing to a certain "low-rank" structure in the potentials. We explore this new class of models by applying the resulting inference methods to neural spike rate estimation and motion-capture joint-angle smoothing tasks.

2.1 Introduction

Graphical models make it easy to compose simple distributions into large, more expressive joint distributions. Unfortunately, in only a small subclass of graphical models is exact computation of marginals and conditionals relatively easy. In particular, while the problem of exact inference in discrete Markov random fields (MRFs) has seen a great deal of attention recently ([Wainwright and Jordan, 2008]), non-Gaussian MRFs defined on more general

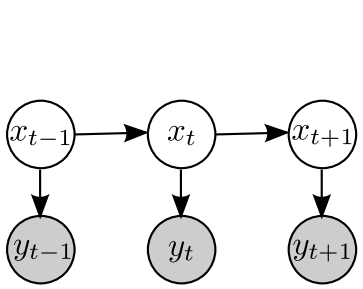
(non-discrete) state-spaces remain a more-or-less open challenge.

As a simple example, consider inference over a chain of dependent probabilities. Such a situation could, for instance, arise when modeling survey responses conducted over many years in which the same yes/no question is asked each year but where the data for some years are missing and of interest. One might want to estimate a population mean latent positive response probability for every year (including those years missing responses) that is expected to vary slowly from year to year. This requires specifying a smoothing prior on a sequence of variables that lie between between zero and one. There are many ways to specify such a smoothing prior, but even in this simple example it is hard to think of models that allow us to compute conditional expectations exactly and efficiently. (For example, the constraints on the latent variables and non-Gaussian likelihood rule out Kalman filtering in a transformed space.)

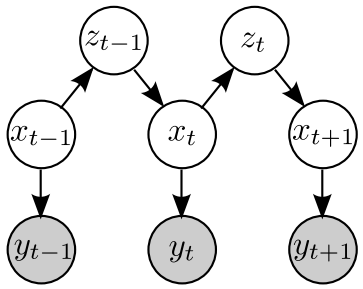
Similar to inferring latent sentiment in a survey response modeling application, one can find other latent variable "smoothing" tasks in fields as diverse as neuroscience and motion capture. In neuroscience, it is of interest to infer the latent probability of spiking – or firing rate – for a neuron given only observations of individual spikes over time. Note that this problem is very similar to the survey response problem above. We show results from "smoothing" neural firing probabilities to demonstrate the exact inference techniques proposed in this paper. We also show an example of smoothing motion capture joint angle time-series data, demonstrating that our exact method is applicable when even classical approximations break down.

The aim of this work is to expand the class of models for which exact inference is computationally feasible, in particular with models of continuous ans structured random variables with non-Gaussian densities. We start by reviewing an auxiliary variable method for introducing Markov chain dependencies between random variables of arbitrary type. We then develop an efficient method for exact inference in a subset of such models, and identify a new class of "low-rank" models in which exact inference is efficient. We perform inference on examples of such models.

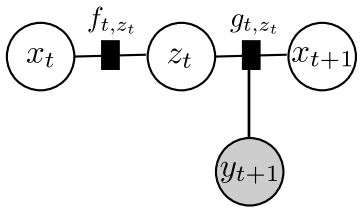
2.2 Related work



(a) Graphical model with "smoothing" dependency between latent variables.



(b) Graphical model with latent "smoothing" dependency induced by auxiliary variables.



(c) Factor graph corresponding to the graphical model in (b).

Figure 2.1

To begin, we first review the work of [Pitt, 2002] and [Pitt and Walker, 2005], who describe an auxiliary variable approach to introducing dependency between random variables of arbitrary types. Refer to Figure 2.1 and consider the sequence of random variables $X = \{x_t\}_{t=1}^T$. (Similarly we use the notation $Y = \{y_t\}_{t=1}^T$ and $Z = \{z_t\}_{t=1}^{T-1}$.) Assume that we would like to bias estimation of the x_t's such that for all values of t, $x_t \approx x_{t+1}$. For now, also assume that we would like the x's in this chain to be marginally identically distributed a priori, i.e. $x_t \sim G_0(x_t)$ for all t (this will be relaxed in later sections). One way to proceed is to require that G_0 is the invariant distribution of a Markov chain with transition kernel $p(x_t|x_{t-1})$, i.e. $G_0(x_t) = \int p(x_t|x_{t-1})G_0(x_{t-1})dx_{t-1}$. This constraint on $p(x_t|x_{t-1})$ is the same as that for any MCMC sampler of G_0; thus $p(x_t|x_{t-1})$ can be any valid sampler transition kernel, e.g. the Metropolis-Hastings transition kernel.

In [Pitt and Walker, 2005] a particular transition kernel based on the Gibbs sampler is

considered. Their clever idea was to form a joint distribution $p(x, z)$ (dropping the subscript notation for the moment), defined as $p(x, z) = p(z|x)G_0(x)$. Clearly, if we Gibbs-sample from this distribution, i.e. sample $z_1 \sim p(z_1|x_1), x_2 \sim p(x_2|z_1), z_2 \sim p(z_2|x_2), \ldots$, then the marginal sequence $x_1, \ldots, x_T$ is marginally distributed as G_0, as desired. One advantage of this approach is that we have a great deal of freedom in our choice of $p(z|x)$. [Pitt and Walker, 2005] and others ([Caron et al., 2007; Gasthaus et al., 2009]) suggest choosing $p(z|x)$ to be conjugate to $G_0(x)$, since this implies that $p(x|z)$ is in the same family as G_0, making sampling more straightforward. In addition, as the amount of information in z about x is increased, neighboring values of x are more closely coupled together. We can also easily incorporate noisy observations y_t from this model (as shown in Figure 2.1): if the likelihood of y_t given x_t is also conjugate to G_0, then $p(x_t|z_{t-1}, y_t)$ remains in the same family as G_0, making conditional Gibbs sampling from $p(X|Y)$ straightforward.

2.3 Low-rank Markov chains

Constructing a Gibbs sampler to sample the x's and z's conditioned on observations (y's in Figure 2.1) is only asymptotically exact in the limit of infinite Gibbs sweeps. What has been overlooked until now (to our knowledge) is that the x's can often themselves be analytically marginalized out, leaving a Markov chain in the z's only, where computation often remains tractable when the z's are discrete random variables with a small state-space. Therefore, in the subset of this class of models in which the z's are discrete random variables, exact inference can be efficiently performed.

To see how this is possible, consider the form of the joint distribution of the graphical model in Figure 2.1b when the z's are discrete random variables. In this case we can write

$$p(X) = p(x_1) \prod_{t=1}^{T-1} \sum_{z_t} p(z_t|x_t)p(x_{t+1}|z_t)$$

where we disregard the observations y_t momentarily for the sake of clarity. To emphasize the primary role of the x's, $p(X)$ can be re-expressed in the following equivalent form

$$p(X) \propto \prod_{t=1}^{T-1} \sum_{z_t=1}^{R_t} f_{t,z_t}(x_t)g_{t,z_t}(x_{t+1}) \tag{2.1}$$

for appropriate functions f_{t,z_t} and g_{t,z_t}, where each sum is a potential coupling neighboring x variables, and where R_t is the size of the state-space of z_t, which we will refer to as the "rank" of the potential, for reasons that will become clear below. (The converse is also true; it is straightforward to show that, given nonnegative f_{t,z_t} and g_{t,z_t}, we can construct corresponding conditionals $p(z_t|x_t)$ and $p(x_{t+1}|z_t)$, although the resulting Markov chain in the x's may be non-stationary). In fact, the conditional distribution $p(X|Y)$ can be expressed in exactly the same form, by absorbing the observation densities $p(y_t|x_t)$ in the f or g terms. In Figure 2.1c we have chosen to include the y_t's in the g factor.

Now, if the x's were discrete random variables, then eq. (2.1) would represent a discrete Markov chain in which the transition matrices are of rank R_t. Recall that exact inference in such a low-rank Markov chain is relatively easy ([Siddiqi et al., 2010]), since the computational complexity of the forward-backward algorithm is dominated by the cost of multiplication by the transition matrix, and multiplication by low-rank matrices is relatively cheap.

The key idea is that, as long as the z's are discrete random variables with small state-space, exact inference on the Markov chain X defined in eq. (2.1) remains tractable. Even in the general (non-discrete X) case, exact inference requires just $O(R^2)$ time (assuming constant $R_t = R$, $t = 1, \ldots, T$), as in a standard low-rank hidden Markov model; here the z's correspond to the latent variables. Consider the partition function of the joint distribution $p(X, Z)$.

$$\begin{aligned}
&\int dX_{1:T} \prod_{t=1}^{T-1} \sum_{z_t=1}^{R_t} f_{t,z_t}(x_t) g_{t,z_t}(x_{t+1}) \\
&\quad = \sum_{z_1=1}^{R_1} \left(\int dx_1 f_{1,z_1}(x_1) \int dx_2 g_{1,z_1}(x_2) \times \right. \\
&\quad \left. \sum_{z_2=1}^{R_2} \left(f_{2,z_2}(x_2) \int dx_3 g_{2,z_2}(x_3) \cdots \right. \right.
\end{aligned}$$

We arrive at the distribution of Z simply by removing the sums over the z's from the partition function. Rearranging the products and integrals above reveals the Markov structure

of Z.

$$\begin{aligned} p(Z) \propto & \int dx_1 f_{1,z_1}(x_1) \int dx_2 g_{1,z_1}(x_2) f_{2,z_2}(x_2) \times \\ & \int dx_3 g_{2,z_2}(x_3) f_{3,z_3}(x_3) \cdots \\ & \int dx_T g_{T-1,z_{T-1}}(x_T) \end{aligned} \tag{2.2}$$

We can use the forward-backward algorithm to compute exact marginals or samples from $p(Z)$; since, given Z, the x's are independent, we can therefore easily compute exact marginals or samples from $p(X)$ as well. To be explicit, expressions for the forward and backward variables are as follows:

$$\begin{aligned} A_1^{(z_1)} &= \int dx_1 f_{1,z_1}(x_1) \\ A_t^{(z_t)} &= \sum_{z_{t-1}}^{R_{t-1}} A_{t-1}^{(z_{t-1})} \int dx_t g_{t-1,z_{t-1}}(x_t) f_{t,z_t}(x_t) \\ B_T^{(z_{T-1})} &= \int dx_T g_{T-1,z_{T-1}}(x_T) \\ B_t^{(z_{t-1})} &= \sum_{z_t}^{R_t} B_{t+1}^{(z_t)} \int dx_t g_{t-1,z_{t-1}}(x_t) f_{t,z_t}(x_t) \end{aligned} \tag{2.3}$$

These are message passing equations ([Bishop, 2006]) and the forward and backward variables can be computed by induction on t. Given these quantities, the marginal distributions of the X become mixture of modes indexed by the z_t:

$$p(x_t) \propto \sum_{z_{t-1}}^{R_{t-1}} A_{t-1}^{(z_{t-1})} \sum_{z_t}^{R_t} B_{t+1}^{(z_t)} g_{t-1,z_{t-1}}(x_t) f_{t,z_t}(x_t)$$

So, if the inner products $\int g_{t-1,i}(x) f_{t,j}(x) dx$ can be evaluated then we can perform exact inference in X (or more generally in X given observations Y) in $O\left(\sum_{t=1}^{T} R_t^2\right)$ time, by the forward-backward algorithm sketched above. (Note that we need only compute these inner products once; these can therefore be pretabulated if necessary before inference begins.) It is straightforward to show that the linear scaling of this inference with T holds for general acyclic Markov random fields (i.e., trees) with potentials of the low-rank form described in eq. (2.1). Moreover, for certain graphs with cycles, the full $p(X)$ or $p(X|Y)$ can be treated efficiently as a weighted sum of trees via the method of conditioning ([Pearl, 1988]).

When applying such a model to data, it will usually not be the case that we know the rank of the potential functions f and g. In this case R has to be estimated from data. This is a standard model selection problem; a Bayesian approach would exploit the marginal likelihood $p(Y|R) = \int p(X|R)p(Y|X)dX$ of the observed data Y given the rank R. This marginal likelihood can be computed directly from our forward recursion (as usual in the context of hidden Markov models ([Rabiner, 1989])); see Fig. 2.2 for an illustration.

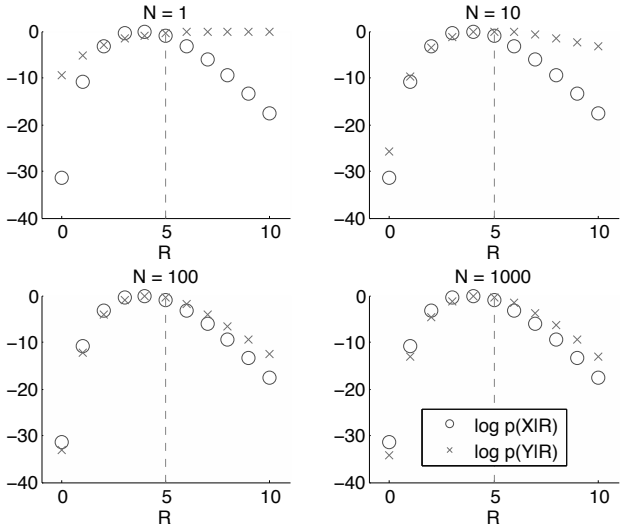


Figure 2.2: The marginal likelihood can be used to estimate the rank of the underlying process X generating data Y. Here a sample X_0 was generated from the beta-binomial time series model $p(X|R)$ (Section 2.4.1) with rank $R = 5$; i.e., $R_t = 5$ for all times t. We plot the log-likelihood (circles) $\log p(X_0|R)$ as a function of R. Then we generate data Y from $p(Y|X_0)$ and use the data to estimate the rank by maximizing the log-likelihood $\log p(Y|R)$ (crosses) as a function of R. As the binomial parameter $N_t \equiv N$ increases, the data Y become more informative about X_0, and $p(Y|R)$ approaches $p(X_0|R)$.

Finally, Z is guaranteed to be a proper Markov chain only if all the inner products over f and g are positive. On the other hand, mathematically there is nothing against performing the recursive inference with the above forward-backward variables when the inner products can be negative, though numerical issues due to cancellation of numbers below machine precision may be a problem in this case. We will stick to nonnegative potentials in this

work.

2.4 Examples

2.4.1 Beta-binomial and Dirichlet-multinomial time series

We now return to the probability-smoothing example we mentioned in the introduction. We consider a time series of binomial distributed data $y_t \sim \text{Binomial}(N_t, x_t)$. If we choose any prior $p(X)$ such that the posterior $p(X|\{N_t\}, Y)$ has the form of eq. (2.1), then exact inference is tractable. For example, we could choose x_t and z_t to have the following simple conjugate Beta-binomial form:

$$\begin{aligned} x_1 &\sim \text{Beta}(\alpha, \beta) \\ z_t|x_t &\sim \text{Binomial}(z_t; R_t, x_t) \\ x_{t+1}|z_t &\sim \text{Beta}(\alpha + z_t, \beta + R_t - z_t) \end{aligned}$$

Thus x_t is marginally $\text{Beta}(\alpha, \beta)$, and the dependence between x_t and x_{t+1} — i.e., the smoothness of the x's as a function of time — is set by R_t: large values of R_t lead to strongly-coupled x_t and x_{t+1}. Eq. (2.1) in this case becomes

$$p(X) = \prod_{t=1}^{T-1} \sum_{z_t=0}^{R_t} a_{z_t} x_t^{z_t} (1 - x_t)^{R_t - z_t} \times$$

$$x_{t+1}^{z_t} (1 - x_{t+1})^{R_t - z_t} \tag{2.4}$$

which we will call the beta-binomial smoother, for the appropriate coefficients a_{z_t}.

We could consider more general priors of the form

$$p(X) \propto \prod_t \sum_{i=0}^{R_t} a_{ti} x_t^{\alpha_i} (1 - x_t)^{\beta_i} x_{t+1}^{\gamma_i} (1 - x_{t+1})^{\delta_i}$$

where α_i, β_i, γ_i, and δ_i are greater than or equal to -1 so that the inner product integrals don't diverge, and $a_{ti} > 0$ for the reasons described above. Given the form of the binomial likelihood, that the posterior $p(X|\{N_t\}, Y)$ will have the same form, but with the constants α_i, β_i, γ_i, and δ_i modified accordingly. Distributions of this form could be considered as tractable conjugate priors for binomial time series data. Note that the necessary inner

products can be computed easily in terms of standard beta functions, and inference proceeds in $O(R^2)$ time, assuming constant $R_t = R$.

Multivariate generalizations are conceptually straightforward: we replace beta distributions with Dirichlets and binomials by multinomials, since by analogy to the beta-binomial model, the Dirichlet is conjugate to the multinomial distribution:

$$\vec{x}_1 \sim \text{Dirichlet}(\vec{\alpha})$$

$$\vec{z}_t|\vec{x}_t \sim \text{Multinomial}(R_t, \vec{x}_t)$$

$$\vec{x}_t|\vec{z}_t \sim \text{Dirichlet}(\vec{\alpha} + \vec{z}_t)$$

$$\vec{y}_t|\vec{x}_t, n_t \sim \text{Multinomial}(\vec{y}_t; \vec{x}_t, n_t)$$

Just as in the beta-binomial case, this defines a sequence of marginally-Dirichlet distributed probabilities x_t, with R_t controlling the smoothness of the state path X. Inference in this case scales quadratically with the total number of possible histograms $\vec{z}_t$ that might be observed.

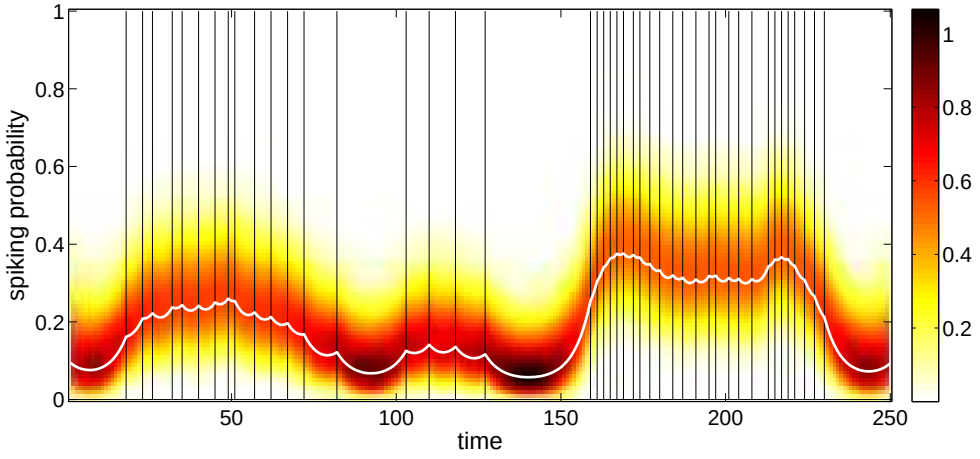


Figure 2.3: The inferred spiking probability density from spike train data assuming a binomial spiking model and the beta-binomial smoother. The black bars are the observed spikes. The solid white line is the inferred mean of the spiking probability. Each time unit is 2 ms.

2.4.2 Smoothing conjugate priors for multinomial data

In many cases one would like a conjugate prior for multinomial data that leads to smooth estimates of the underlying probabilities. In the preceding example, we constructed a conjugate prior for count data that has smooth and nonnegative sample paths. If we further constrain these sample paths to sum to one, then we could interpret X as a discrete probability distribution; it is easy to see that the resulting smoothing prior $p(X)$ is conjugate to multinomial data, due to the completely factorized form of the multinomial likelihood. However, it is not immediately clear how to exploit the model's low-rank structure to perform inference in a tractable way, since the constraint that the components of X sum to one breaks the tree structure of the graphical model.

One approach is to transform to a larger state-space, $x_t \to q_t = (x_t\ s_t)$, where s_t denotes the cumulative sum $s_t = x_1 + x_2 + \cdots + x_t$. This leads to a Markov prior on the augmented state variable Q of the form

$$\begin{aligned} p(Q) \propto \delta(s_1 = x_1) x_1^{\nu_1 - 1} \sum_{k_1}^{R} a_{k_1} x_1^{k_{1,1}} x_2^{k_{1,2}} \times \\ \delta(s_1 = s_2 - x_2) x_2^{\nu_2 - 1} \sum_{k_2}^{R} a_{k_2} x_2^{k_{2,1}} x_3^{k_{2,2}} \times \\ \cdots \delta(s_{T-2} = s_{T-1} - x_{T-1}) x_{T-1}^{\nu_{T-1} - 1} \times \\ \sum_{k_{T-1}}^{R} a_{k_{T-1}} x_{T-1}^{k_{T-1,1}} x_T^{k_{T-1,2}} \times \\ \delta(s_{T-1} = s_T - x_T) x_T^{\nu_T} \delta(s_T = 1) \end{aligned} \tag{2.5}$$

As outlined in greater detail in the appendix, we can perform forward-backward inference on this density by recursively integrating the above density, over all the x_i and the s_i, to compute the normalization constant, and then rearranging the summations into the form of a sum-product algorithm. The resulting inference algorithm requires $O(R^2T^2)$ storage and $O(R^6T^2)$ processing time.

2.4.3 Phase data

So far our random variables have lived in convex subsets of vector spaces; standard approximation methods (e.g., Laplace approximation ([Kass and Raftery, 1995]) or expectation propagation ([Minka, 2001])) can often be invoked to perform approximate inference in these settings. However, our method may be applied on more general state-spaces, where these classical approximations break down. As a concrete example, consider a time-series of phase variables (angles). The von Mises distribution

$$p(x_t|\mu_t, \kappa_t) \propto e^{\kappa_t \cos(x_t - \mu_t)}$$

(with mean and concentration parameters μ_t and κ_t) is popular for modeling one-dimensional angular data, largely because the necessary normalization factors can be computed easily, and furthermore this model has the convenient feature that, like the normal density, it is conjugate to itself ([Gelman et al., 2003]). As in our previous examples, this univariate distribution can be augmented to tractably model smoothed time-series data. For instance, we could take

$$p(X) \propto \prod_t \sum_{i=0}^{R} e^{\frac{R}{2}\cos\left(x_t - \frac{2\pi i}{R+1}\right)} e^{\frac{R}{2}\cos\left(x_{t+1} - \frac{2\pi i}{R+1}\right)} \tag{2.6}$$

This acts as a smoothing prior, since at each time t, for each corresponding pairwise potential, each of the terms in the sum over i is a unimodal function peaked at $x_t = x_{t+1} = \frac{2\pi i}{R+1}$. That is, each term contributes a bump along the diagonal, and therefore the sum over i corresponds to a nearly-diagonal transition matrix, i.e. to a smoothing prior. Larger values of R lead to smoother sample paths in X. Inference proceeds as in the previous examples; if the observations y_t also have von Mises densities given x_t (as in the example application discussed in the next section), then the necessary inner products can be computed easily in terms of Bessel functions.

As in the Dirichlet-multinomial case, extensions to multivariate phase data are conceptually straightforward (the von Mises-Fisher density generalizes the univariate von Mises density ([Mardia and Jupp, 2000]); see [Cadieu and Koepsell, 2010] for another generalization). We will describe another generalization, to oscillatory or narrowband time series data, below.

2.5 Experiments

We began by analyzing some simple neural spike train data (http://neurotheory.columbia.edu/~larry/book/exercises.html) using the beta-binomial smoother. A segment of a spike train in which each time unit represents 2 ms was obtained. The spikes (the binary observations $\{y_t\}$) were modeled as draws from a binomial distribution with time-varying probability x_t. The smoother of eq. (2.4) was used with $\alpha = \beta = 1$, setting the a priori marginals to be uniform distributions. We used $R = 100$, which leads to a prior autocorrelation time of approximately 60 ms. The forward-backward algorithm was run to infer the distribution over x_t as a function of time as shown in Figure 2.3. The marginal mean varies smoothly over time, rising during times of higher spike rates.

We also performed some basic comparisons to Gibbs sampling. The Gibbs sampler is the standard approach to computation in this type of model, but as emphasized above it only leads to approximate solutions, whereas the marginalized forward-backward approach we have introduced here provides exact results. The basic result, shown in Figure 2.4 is unsurprising: many Gibbs sweeps are required to achieve a certain error level, particularly in cases where the sample paths from the conditional distribution $p(X|Y, R)$ are strongly coupled.

Next we turned to a dataset involving phase variables. We analyzed joint articulation motion capture data from the CMU Graphics Lab Motion Capture Database (http://mocap.cs.cmu.edu/search.php?subjectnumber=13&trinum=9). A time-series of angles of extension of the right radius of a man drinking from a bottle of soda was analyzed. This motion was modeled with the von Mises smoother of eq. (2.6) with $R = 20$ and $\kappa = 2$. The observations y_t were modeled as von Mises draws with mean x_t. The forward-backward algorithm smoothed the data effectively and allowed for appropriate inference in the presence of missing data, as illustrated in Fig. 2.5.

Conceptually, we are applying a rather simple state-space model to this data, with the true underlying angle (the hidden state variable) modeled as $x_{t+1} = x_t + \epsilon_t$, and the observation modeled as $y_t = x_t + \eta_t$ for appropriate noise terms ϵ_t and η_t. This state-space viewpoint suggests some natural further generalizations. For example, if we let $x_{t+1} = x_t + 2\pi\omega + \epsilon_t$, then x_t could model a narrowband signal with dominant frequency ω. Our

inference methods can be applied in a straightforward manner to this oscillatory model, and may therefore be useful in a number of potential applications, e.g. the analysis of noisy electroencephalography data, or in the acoustic applications described in [Turner and Sahani, 2011].

2.6 Discussion

We have introduced a class of "low-rank" models for continuous-valued data in which exact inference is possible by efficient forward-backward methods. These exactly-solvable models are perhaps of most interest in cases where standard approximation methods (e.g., expectation propagation or Laplace approximation) are unreliable, such as the application to circular data time series discussed in section 2.4.3. Even in less "exotic" cases, such as the beta-binomial model discussed in section 2.4.1, classical methods based on Gibbs sampling can mix slowly (c.f. Fig. 2.4), making the exact sampler introduced here more attractive. (More generally, of course, there is significant value in exact, not approximate, inference methods: in mission-critical applications, for example, it is essential to have methods that are guaranteed to return the correct answer 100% of the time.) Thus we hope that these low-rank models might prove useful in a wide range of applications.

Directions for future theoretical research include connections with recent work on inference in reproducing kernel Hilbert spaces (RKHSs) by [Song et al., 2010b] and [Song et al., 2010a]. The latter describe an approximate RKHS inference method in tree-structured graphical models, e.g. over non-Gaussian continuous variables. They perform belief propagation by operations on messages represented in a RKHS. Their approximate inference algorithm takes as input samples from the variables and estimates the necessary operators. It could be fruitful to explore connections between such a program and the methods we present here.

Our models also bear resemblance to the Reduced-Rank Hidden Markov Models (RR-HMMs) proposed by [Siddiqi et al., 2010]. These are n-state hidden Markov models with rank $k < n$ transition matrices. Probabilities in RR-HMMs can be represented in terms of $k \times k$ matrices, and inference can be done in $O(k^2)$ time. Our method exploits a similar prop-

erty of the models it treats, reformulating transitions between continuous state-spaces (not only between high-dimensional discrete state-spaces) in terms of low-dimensional discrete spaces.

Further, there are a number of models that include inference in a large number of chains of dependent, constrained random variables for which our exact inference approach might not only improve inference but may result in significant computational savings. One example is the generalized Polya-urn dependent Dirichlet process (GPU-DDP) mixture ([Caron et al., 2007]). The GPU-DDP models time series observations as being draw from a time-dependent Dirichlet mixture. The latent parameters of the mixture components are allowed to change over time, but must be constrained in the same way that the auxiliary variable random walk of [Pitt, 2002] constrains the latent sample paths in this paper. Inference in GPU-DDP mixtures is hard, suffering from slow mixing and high computational complexity, particularly in the low sample count, high-rank domain in which our exact inference approach excels. Applying our inference procedure to GPU-DDP inference could result in substantial improvements.

2.7 Appendix

In section 2.4.2 we describe a polynomial time smoother for multinomial data confined to the unit simplex. Here we derive the forward variables of this smoother, which we need to perform efficient inference. The necessary backward variables may be derived following the same integration steps but starting from time T and proceeding backward.

Define $s_0 \equiv 0$, and let k_i be the multi-index $[k_{i1}, k_{i2}]$, over which the sum $\sum_{k_i} x_i^{k_{i1}} x_{i+1}^{k_{i2}}$ couples x_i to x_{i+1} (in addition to the implicit coupling by the simplex constraint). Let $\{a_k(t)\} = \{a_k\}$ be a set of coefficients, indexed by multi-index k, that we assume to be constant over time, merely for notational convenience. Pushing all sums as far to the right as possible, and defining $\delta_i \equiv \delta(s_i = s_{i+1} - x_{i+1})$, the joint density in the expanded

state-space is

$$\begin{aligned} p(Q) = &\sum_{k_1}^{R} a_{k_1} x_1^{\nu_1+k_{1,1}-1}\delta_0 \times \\ &\sum_{k_2}^{R} a_{k_2} x_2^{\nu_2+k_{1,2}+k_{2,1}-1}\delta_1 \times \cdots \\ &\sum_{k_{T-1}}^{R} a_{k_{T-1}} x_{T-1}^{\nu_{T-1}+k_{T-2,2}+k_{T-1,1}-1}\delta_{T-2} \times \\ &x_T^{\nu_T+k_{T-1,2}-1}\delta_{T-1}\delta(s_T = 1) \end{aligned}$$

For the purpose of computing the forward variables, the multinomial exponents $\{\nu_i - 1\}$ may be omitted and considered absorbed by the $k_{i,j}$, so as to further simplify the notation.

We compute the normalization constant of the joint distribution:

$$\begin{aligned} \mathcal{Z} &= \int_0^1 ds_1 \int_0^{s_1} dx_1 \cdots \int_0^1 ds_T \int_0^{s_T} dx_T \, p(Q) \\ &= \int_0^1 ds_1 \int_0^{s_1} dx_1 \sum_{k_1}^{R} a_{k_1} x_1^{k_{1,1}} \delta_0 \times \\ &\int_0^1 ds_2 \int_0^{s_2} dx_2 \sum_{k_2}^{R} a_{k_2} x_2^{k_{1,2}+k_{2,1}} \delta_1 \times \cdots \end{aligned}$$

To compute $\mathcal{Z}$, we integrate from left to right, first integrating dx_1, then ds_1, then dx_2, then ds_2, and so on until ds_T. Even after only the first five integrations, a pattern begins to emerge:

$$\begin{aligned} \mathcal{Z} = &\sum_{k_1} a_{k_1} \int_0^1 ds_1 (s_1 - s_0)^{k_{1,1}} \times \\ &\int_0^1 ds_2 \int_0^{s_2} dx_2 \sum_{k_2} a_{k_2} x_2^{k_{1,2}+k_{2,1}} \delta_1 \times \cdots \\ = &\sum_{k_1} a_{k_1} \int_0^1 ds_2 \int_0^{s_2} dx_2 (s_2 - x_2)^{k_{1,1}} \times \\ &\sum_{k_2} a_{k_2} x_2^{k_{1,2}+k_{2,1}} \times \cdots \\ = &\sum_{k_1} a_{k_1} \sum_{k_2} a_{k_2} \mathrm{B}(k_{1,2} + k_{2,1}, k_{1,1}) \times \end{aligned}$$

$$\int_0^1 ds_2 s_2^{k_{1,1}+k_{1,2}+k_{2,1}} \times$$
$$\int_0^1 ds_3 \int_0^{s_3} dx_3 \sum_{k_3} a_{k_3} x_3^{k_{2,2}+k_{3,1}} \delta_2 \times \cdots$$

$$= \sum_{k_1} a_{k_1} \sum_{k_2} a_{k_2} \mathrm{B}(k_{1,2}+k_{2,1}, k_{1,1}) \times$$
$$\int_0^1 ds_3 \int_0^{s_3} dx_3 (s_3 - x_3)^{k_{1,1}+k_{1,2}+k_{2,1}} \times$$
$$\sum_{k_3} a_{k_3} x_3^{k_{2,2}+k_{3,1}} \times \cdots$$

$$= \sum_{k_1} a_{k_1} \sum_{k_2} a_{k_2} \mathrm{B}(k_{1,2}+k_{2,1}, k_{1,1}) \times$$
$$\sum_{k_3} a_{k_3} \mathrm{B}(k_{2,2}+k_{3,1}, k_{1,1}+k_{1,2}+k_{2,2}) \times$$
$$\int ds_3 s_3^{k_{1,1}+k_{1,2}+k_{2,1}+k_{2,2}+k_{3,1}} \times$$
$$\int_0^1 ds_4 \int_0^{s_4} dx_4 \sum_{k_4} a_{k_4} x_4^{k_{3,2}+k_{4,1}} \delta_3 \times \cdots$$

The first equality results from integration with respect to x_1, which removes the delta function $\delta_0 = \delta(s_0 = s_1 - x_1) = \delta(x_1 = s_1)$, leaving $(s_1 - s_0)^{k_{1,1}}$ in place of $x_1^{k_{1,1}}$. The second equality results from integration with respect to s_1, which removes the delta function $\delta_1 = \delta(s_1 = s_2 - x_2)$, which replaces $(s_1 - s_0)^{k_{1,1}} = s_1^{k_{1,1}}$ with $(s_2 - x_2)^{k_{1,1}}$. Next we integrate with respect to x_2, which proceeds by the following change of variables:

$$\int_0^s dx (s-x)^{\alpha-1} x^{\beta-1} \stackrel{u=x/s}{=}$$
$$\int_0^1 du (s(1-u))^{\alpha-1} (su)^{\beta-1} s =$$
$$s^{\alpha+\beta-1} \int_0^1 (1-u)^{\alpha-1} u^{\beta-1} = s^{\alpha+\beta-1} \mathrm{B}(\alpha, \beta)$$

where $\mathrm{B}(\alpha, \beta)$ is the beta function. The next integration with respect to s_2 removes the delta function $\delta_2 = \delta(s_2 = s_3 - x_3)$, resulting in the power of $(s_3 - x_3)$. The last equality is a result of the same change of variables when integrating with respect to x_3, yielding another beta function, and leaving the last line of the expression identical to the last line of the expression two steps before, except that the "time" indices have all increased by one.

To condense this expression, define $k_{0,2} \equiv k_{T,1} \equiv 0$, and $K_i \equiv \sum_{j=1}^{i} k_{j-1,2} + k_{j,1}$. Lastly, define $b(k_{1:n}) \equiv \mathrm{B}(k_{n-1,2} + k_{n,1}, K_{n-1})$. Continuing with the integration from left to right, we find an expression for the normalization constant as the following nested sum:

$$\begin{aligned}\mathcal{Z} = \sum_{k_1}^{R} a_{k_1} \left(\sum_{k_2}^{R} a_{k_2} b(k_{1:2}) \left(\sum_{k_3}^{R} a_{k_3} b(k_{1:3}) \times \cdots \right.\right. \\ \left.\left(\sum_{k_{T-2}}^{R} a_{k_{T-2}} b(k_{1:T-2}) \left(\sum_{k_{T-1}}^{R} a_{k_{T-1}} b(k_{1:T-1}) \ \times \right.\right.\right. \\ \left.\left.\left. b(k_{1:T}) \right) \right) \cdots \right) \right)\end{aligned}$$

This is not in sum-product form because, e.g., $b(k_{1:T})$ depends on all the indices of summation. We may put this into sum-product form as follows. However, we can rearrange this summation into the form of a tractable sum-product algorithm as follows. Each sum over k_i is first a sum over $k_{i,1} \in \{0, \cdots, R\}$, and then a sum over $k_{i,2} \in \{0, \cdots, R\}$. The sums are in the order $k_{1,1}$, $k_{1,2}$, $k_{2,1}$, $k_{2,2}$, and so on. Now, notice that every set of values $\{k_i\}_{i=1}^{T} = \{[k_{i,1}, k_{i,2}]\}_{i=1}^{T}$ corresponds uniquely to a set of values $\{[k_{i,1}, K_i]\}_{i=1}^{T}$, with K_i as defined above, and vice versa. Then we may sum over values of the latter quantity in the order K_T, $k_{T-1,1}$, K_{T-1}, $k_{T-2,1}$, and so on, instead of summing over values of k_i, and obtain the same result. That is, we treat the K_i as sum indices instead of explicitly summing over the $k_{i,2}$.

To sum over all possible values of this second set of indices, the values of the indices must be constrained to be compatible. For instance, $k_{1,1} = 1$ is not compatible with $K_2 = 0$, because $K_2 = k_{1,1} + k_{1,2} + k_{2,1} \geq k_{1,1}$. This compatibility requirement manifests in the upper and lower bounds of the sums over these indices in eq. (2.7). Consider the sum over $k_{1,1}$, given some values for $k_{2,1}$ and K_2 (which come earlier in the multiple sum). *A priori*, $k_{1,1} \in \{0, \cdots, R\}$. However, $k_{1,1} = K_2 - k_{2,1} - k_{1,2} \leq K_2 - k_{2,1}$ and $k_{1,1} = K_2 - k_{2,1} - k_{1,2} \geq K_2 - k_{2,1} - R$, so we have $k_{1,1} \in \{\max\{0, K_2 - k_{2,1} - R\}, \cdots, \min\{R, K_2 - k_{2,1}\}\}$. These are the values of $k_{1,1}$ that are to be summed over, i.e. that are compatible with the values of previously specified indices. More generally,

$$\begin{aligned}k_{i,1} &= K_{i+1} - k_{i+1,1} - K_{i-1} - k_{i-1,2} - k_{i,2} \\ &\geq K_{i+1} - k_{i+1,1} - (2i-1)R\end{aligned}$$

$$
\begin{aligned}
k_{i,1} &= K_{i+1} - k_{i+1,1} - K_{i-1} - k_{i-1,2} - k_{i,2} \\
&\leq K_{i+1} - k_{i+1,1}
\end{aligned}
$$

$$
0 \leq k_{i,1} \leq R
$$

$$
\begin{aligned}
k_{i,1} \in \{&\max\{0, K_{i+1} - k_{i+1,1} - (2i-1)R\}, \cdots, \\
&\min\{R, K_{i+1} - k_{i+1,1}\}\}
\end{aligned}
$$

Similarly,

$$
\begin{aligned}
K_i &= K_{i+1} - k_{i+1,1} - k_{i,2} \\
&\geq K_{i+1} - k_{i+1,1} - R
\end{aligned}
$$

$$
\begin{aligned}
K_i &= K_{i+1} - k_{i+1,1} - k_{i,2} \\
&\leq K_{i+1} - k_{i+1,1}
\end{aligned}
$$

$$
K_i \geq k_{i,1}
$$

$$
\begin{aligned}
K_i \in \{&\max\{k_{i,1}, K_{i+1} - k_{i+1,1} - R\}, \cdots, \\
&K_{i+1} - k_{i+1,1}\}
\end{aligned}
$$

We redefine $b(k_{1:n}) = \mathrm{B}(k_{n-1,2} + k_{n,1}, K_{n-1})$ as the same quantity in terms of the new indices of summation, $b(K_{n-1}, K_n) \equiv \mathrm{B}(K_n - K_{n-1}, K_{n-1})$. Putting this all together we can write the normalization constant in the form of a sum-product algorithm, arranging the

sums in the prescribed order and pushing all factors as far to the left as possible:

$$\begin{aligned}
\mathcal{Z} = \sum_{K_T=0}^{R(2T-2)} \Bigg(& \sum_{k_{T-1,1}=\max\{0,K_T-(2(T-2)-1)R\}}^{\min\{R,K_T\}} \\
& \sum_{K_{T-1}=\max\{k_{T-1,1},K_T-R\}}^{K_T} \\
& a_{[k_{T-1,1},K_T-K_{T-1}]} b(K_{T-1},K_T)\,(\cdots(\\
& \sum_{k_{3,1}=\max\{0,K_4-k_{4,1}-5R\}}^{\min\{R,K_4-k_{4,1}\}} \sum_{K_3=\max\{k_{3,1},K_4-k_{4,1}-R\}}^{K_4-k_{4,1}} \\
& a_{[k_{3,1},K_4-K_3-k_{4,1}]} b(K_3,K_4)\,(\\
& \sum_{k_{2,1}=\max\{0,K_3-k_{3,1}-3R\}}^{\min\{R,K_3-k_{3,1}\}} \sum_{K_2=\max\{k_{2,1},K_3-k_{3,1}-R\}}^{K_3-k_{3,1}} \\
& a_{[k_{2,1},K_3-K_2-k_{3,1}]} b(K_2,K_3)\,(\\
& \sum_{k_{1,1}=K_1=\max\{0,K_2-k_{2,1}-R\}}^{\min\{R,K_2-k_{2,1}\}} \\
& a_{[k_{1,1},K_2-K_1-k_{2,1}]} b(K_1,K_2))))\cdots)\Bigg)
\end{aligned} \tag{2.7}$$

Note that since we are no longer explicitly summing over $k_{i,2}$, it has been replaced by $K_{i+1} - K_i - k_{i+1,1}$ in the second element of the multi-index indexing the coefficients $\{a_k\}$. The forward variables $A^{(i,j)}$, then, can be immediately read off as follows. The first and

second superscripts index values of $k_{t,1}$ and K_t, respectively:

$$A_2^{(i,j)} = \sum_{k=\max\{0,j-i-R\}}^{\min\{R,j-i\}} a_{[k,j-k-i]} \mathrm{B}(j-k,k)$$

$$i \in \{0,\cdots,R\}, j \in \{i,\cdots,i+2R\}$$

$$A_t^{(i,j)} = \sum_{k=\max\{0,j-i-(2t-3)R\}}^{\min\{R,j-i\}} \sum_{l=\max\{k,j-i-R\}}^{j-i}$$

$$a_{[k,j-l-i]} \mathrm{B}(j-l,l) A_{t-1}^{(k,l)}$$

$$i \in \{0,\cdots,R\}, j \in \{i,\cdots,i+2(t-1)R\}$$

$$A_T^{(j)} = \sum_{k=\max\{0,j-(2t-4)R\}}^{\min\{R,j\}} \sum_{l=\max\{k,j-R\}}^{j}$$

$$a_{[k,j-l]} \mathrm{B}(j-l,l) A_{T-1}^{(k,l)}$$

$$j \in \{0,\cdots,2(T-1)R\}$$

and $\mathcal{Z} = \sum_{i=0}^{2(T-1)R} A_T^{(i)}$. Similarly we can derive backward variables $C_t^{(i,j)}$, where the first superscript indexes $k_{t-1,2}$ and the second indexes $L_t \equiv \sum_{i=t}^{T} k_{i-1,2} + k_{i,1}$. Marginal quantities can be computed readily. The singleton marginal density is

$$p(x_t) = \frac{1}{\mathcal{Z}} \sum_{i=0}^{R} \sum_{j=i}^{i+2(t-2)R} \sum_{k=0}^{R} \sum_{l=k}^{k+2(T-1-t)R} \tag{2.8}$$
$$A_{t-1}^{(i,j)} C_{t+1}^{(k,l)} \mathrm{B}(j,l) \sum_{k_{t-1,2}=0}^{R} \sum_{k_{t,1}=0}^{R}$$
$$a_{[i,k_{t-1,2}]} a_{[k_{t,1},k]} x_t^{k_{t-1,2}+k_{t,1}} (1-x_t)^{j+l}$$

Therefore this method requires $O(R^2T^2)$ storage for the forward and backward variables, and $O(R^6T^2)$ processing time to compute marginals. For processing time, one factor of T comes from the number of marginals to compute. The other factor comes from the number of forward variables for each time, which increases linearly with T, manifesting in the limits of the sums over j and l in eq. (2.8).

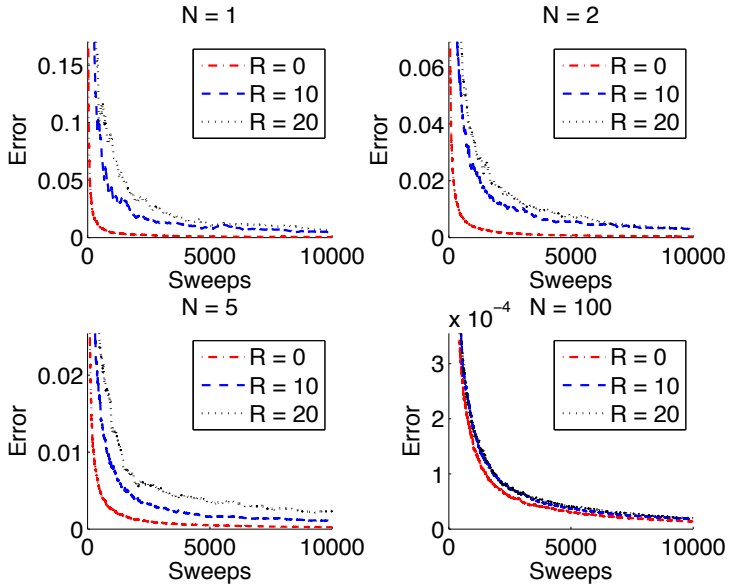


Figure 2.4: For different amounts N of data, the marginal means of a Markov chain X of length $T = 100$ were computed both exactly and approximately by Gibbs sampling, using the beta-binomial smoother. Here we plot the root mean square error per time step of the Gibbs solution with respect to the exact solution as a function of the number of Gibbs sweeps. For each value N of the binomial count parameter we plot this curve for three values of the rank R. Each curve is the median of 25 traces, each the average of 10 independent runs of the Gibbs sampler. Each of the 25 traces corresponds to different randomly generated input data from $p(Y|R)$. The sampler was initialized with each x_t drawn independently from the marginal prior distribution, $p(x_t) = Uniform([0, 1])$. The Gibbs estimates converge most quickly when $p(X|Y, R)$ is most uncoupled, that is, when R is small and/or N is large; when R is large or N is small the Gibbs error requires many sweeps to shrink towards zero.

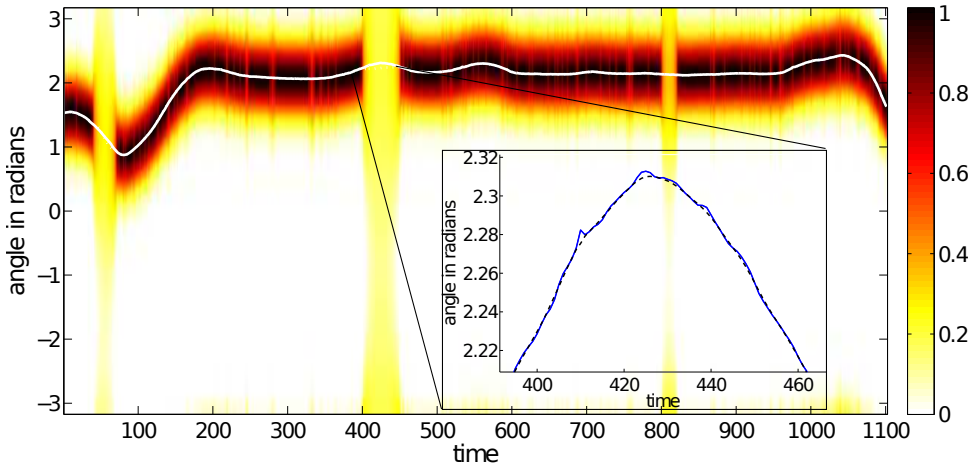


Figure 2.5: The inferred probability density of angle in motion capture data. The solid white line is the observed signal. The dotted white line, mostly obscured by the solid white line, is the inferred mean. The colorbar indicates the inferred posterior. Bands appear in intervals where the observations are suppressed. They are tapered because, deeper within the band, distant observations are less informative of the density, which is therefore nearly uniform. Inset: Inference with no data held out. The solid line is the observed signal, the dashed line the inferred mean. Much of the noise in the signal has been smoothed. Each time unit is 2 ms.

Chapter 3

Information loss in spike trains due to spike time jitter and ambiguity of neuronal identity

3.1 Abstract

We investigate Bayesian methods for optimal decoding of noisy or incompletely-observed spike trains. Information about neural identity or temporal resolution may be lost during spike detection and sorting, or spike times measured near the soma may be corrupted with noise due to stochastic membrane channel effects in the axon. We focus on neural encoding models in which the (discrete) neural state evolves according to stimulus-dependent Markovian dynamics. Such models are sufficiently flexible that we may incorporate realistic stimulus encoding and spiking dynamics, but nonetheless permit exact computation via efficient hidden Markov model forward-backward methods. We analyze two types of signal degradation. First, we quantify the information lost due to jitter or downsampling in the spike-times. Second, we quantify the information lost when knowledge of the identities of different spiking neurons is corrupted. In each case the methods introduced here make it possible to quantify the dependence of the information loss on biophysical parameters such as firing rate, spike jitter amplitude, spike observation noise, etc. In particular, decoders

that model the probability distribution of spike-neuron assignments significantly outperform decoders that use only the most likely spike assignments, and are ignorant of the posterior spike assignment uncertainty.

3.2 Introduction

Bayesian decoding of spike trains has received a great deal of previous attention; see e.g. [Pillow et al., 2011] for a recent review. Bayesian decoding methods are particularly appealing because they are optimal in principle (assuming that the "encoding model" — the probabilistic model that describes how information is encoded in the spike trains — is correctly specified), and also because prior knowledge can be explicitly incorporated into the Bayesian model [Kass et al., 2005].

Much of the previous neural decoding literature has assumed that the spike trains to be decoded have been observed completely and noiselessly. Of course, in practice this is rarely the case. For example, information about neural identity or correlations may be lost during spike sorting [Lewicki, 1998; Hill et al., 2011], or spike times measured near the soma may be corrupted with noise due to stochastic membrane channel effects in the axon [Aldworth et al., 2005; Faisal et al., 2005; Faisal and Laughlin, 2007]. The recent rise in popularity of optical (e.g., calcium-based) methods for spike detection (which typically offer significantly less signal resolution than electrical recording methods) have made these issues more pressing [Cossart et al., 2003; Ohki et al., 2005]; see, e.g., [Mishchenko et al., 2011] for a recent related discussion.

The main contribution of this work is a general framework, using flexible spiking models of populations of neurons, for computationally tractable Bayesian spike train decoding when spike trains are corrupted by either the presence of spike time jitter or spike sorting identity errors. Our goal is to obtain a better analytical and computational understanding of the impact of these spike corruptions on the optimal decoder. Our approach makes heavy use of well-known efficient inference algorithms for hidden Markov models.

Related questions have been previously investigated using simple Poisson neuron models [Aldworth et al., 2005; Gollisch, 2006; Ventura, 2008b]. In this paper we extend these anal-

yses to a more realistic and flexible class of spiking models. We focus on a Markovian model of spiking dynamics that is similar to those treated in [Herbst et al., 2008; Toyoizumi et al., 2009; Calabrese and Paninski, 2011; Escola et al., 2011; Nossenson and Messer, 2011], and which is closely related to the spike-response/generalized-linear model framework that has become popular in the recent computational and statistical neuroscience literature [Truccolo et al., 2005; Paninski et al., 2007]. This model class is sufficiently flexible that we may incorporate realistic stimulus encoding and spiking dynamics, but nonetheless permits exact computation via efficient forward-backward methods familiar from the theory of hidden Markov models.

Any such Bayesian approach must specify a prior distribution on the signals to be decoded. In the context of this Markovian neural encoding model, we note that the broad class of "low-rank" state-space models recently introduced in [Smith et al., 2012] provides a convenient set of conjugate priors [Casella and Berger, 2001] in our setting, implying that the posterior marginal distributions of stimuli given the model's sufficient statistics can be computed exactly, further enabling efficient computation.

We focus here on two major mechanisms of spike train corruption. First, we examine the impact of errors in the timing of each observed spike on decoder performance. Second, we quantify the loss of decoding efficiency when knowledge of the identities of the observed neurons is discarded or corrupted. A concrete example of the latter problem involves spike sorting in low-SNR regimes, where overlaps in spike clusters can lead to errors or excessive uncertainty in the identity of the neuron contributing any observed spike. In each case, our methods allow us to quantify and compute the loss in decoding performance efficiently over a range of parameter values, contributing to a more systematic understanding of the importance of these effects.

The paper is organized as follows. We first define more specifically the models of spike train corruption that we will consider, along with the relevant model assumptions. Then we will describe how to compute the resulting Bayesian decoders, given spike train data which has been corrupted by these mechanisms. In each case, the decoder involves the execution of a two-stage Gibbs sampler [Robert and Casella, 2005] that we will describe in detail. We will briefly describe a class of stimulus priors for which our methods are made particularly

efficient by "Rao-Blackwellization" of the Gibbs sampler, in which the sample average is replaced with an estimator with lower variance [Robert and Casella, 2005]. Finally, we will present the results of some analyses of simulated data, where we can compare directly to Bayesian decoders given uncorrupted spike train data, and close with a discussion of some open directions for future research.

3.3 Sources of information loss

For simplicity, we assume that each observed neuron responds conditionally independently to the stimulus (Fig. 3.1). We will focus on Bayesian reconstruction of stimuli from spike trains corrupted by spike-time jitter and neuron identity loss. In both cases, we assume that the state of each neuron evolves according to some Markovian dynamics, passing through single-neuron states $Q = \{q_t\}$, where q_t is the state of the neuron at time t. Some or all of the transition probabilities may be functions of the stimulus $X = \{x_t\}$, where x_t is the value of the stimulus at time t. (As usual, the X may be viewed more generally as some filtered or transformed representation of the physical stimulus, or may represent some more abstract covariate that the neural activity depends upon.) We assume that time has been discretized into equal-length bins that are sufficiently small that a given neuron can fire at most once within each bin.

3.3.1 Spike time jitter

Biophysically, sources of the variability in spike-times include stochasticity in the activity of ion channels and in synaptic transmission, or in the timing of spike detection, particularly in optical recordings. (Note that temporal downsampling can also be interpreted as a form of temporal noise, once the spike times are upsampled back to their original resolution.) The input to the decoder, then, may be a set of spike trains $Y = \{y_t\},\ y_t \in \{0, 1\}$, jittered from the true spike-time data D. In devising a decoding algorithm, one might ignore these sources of noise and take the observed spike-times as the actual times. Alternatively, one may perform inference directly on the temporally corrupted signal, explicitly modeling the sources of temporal noise.

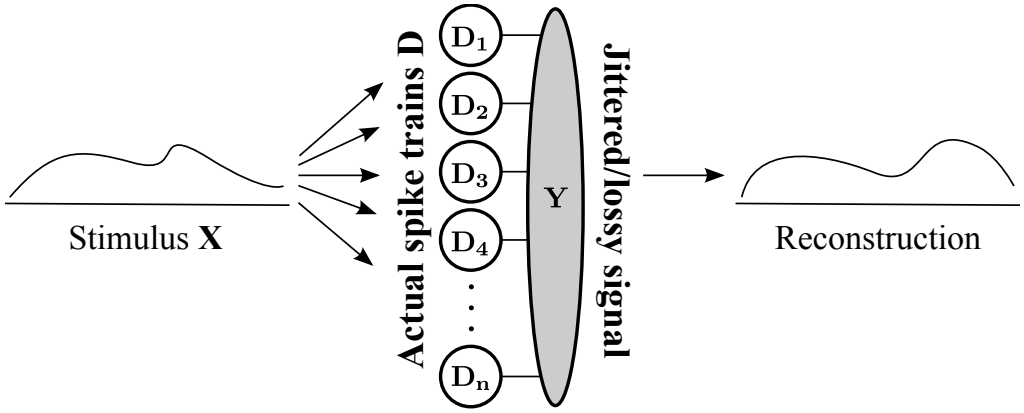


Figure 3.1: Problem schematic. Neurons are exposed to a common stimulus X. They generate spike trains D which are then degraded by sources of noise. By Y we denote the degraded signal, which is the input to a decoder that reconstructs the stimulus. The spike trains D are assumed to be conditionally independent given the stimulus X.

The importance of spike-time noise has been recognized previously; for example, [Aldworth et al., 2005] describe an iterative algorithm for dejittering spike trains by aligning single stimuli to find the most likely stimulus to have preceded a jittered spike. Our approach differs by attempting to reconstruct the entire stimulus given a jittered full spike train, instead of a single spike. In addition, [Gollisch, 2006] introduces an iterative expectation-maximization (EM) algorithm that improves receptive field estimates by explicitly incorporating spike-time jitter into a linear-nonlinear Poisson (LNP) model. By contrast, we will focus on stimulus decoding, rather than the estimation of encoding models; in addition, our approach is able to handle more general non-Poisson spike generation.

3.3.2 Spike identity loss

A key challenge in neural recording is the problem of spike sorting: correctly assigning each spike to the neuron that generated the spike, given noisy extracellular voltage or optical recordings [Lewicki, 1998; Hill et al., 2011]. Any single electrode (or optical pixel) will often record the activity of more than one neuron, in addition to any background noise. An optimal decoder therefore needs to keep track of the posterior uncertainty corresponding to the assignment of each spike, as previously emphasized, e.g., by [Wood and Black, 2008;

Ventura, 2008b; Ventura, 2008a], in the context of spike trains which can be modeled as inhomogeneous Poisson processes. (See also [Chen et al., 2012], who like [Ventura, 2008b] emphasize the importance of retaining as much information as possible in the raw extracellular voltage signal, and not discarding the "unsortable" spikes.) Again, a key extension here is to generalize to the non-Poisson setting.

3.4 Model assumptions

We model the dynamics of the neurons in the observed populations in terms of discrete-time Markov chains, similar to the models treated, e.g., in [Sahani, 1999; Herbst et al., 2008; Calabrese and Paninski, 2011; Escola et al., 2011]. These models are sufficiently flexible to handle refractoriness, burstiness, adaptation, and other aspects of spike train dynamics while at the same time remain highly computationally tractable, as we will describe in further depth below.

Fig. 3.2 illustrates a simple example of a Markovian model neuron with three states. In this model, spiking occurs in the state (1). States (2) and (3) are non-spiking states, and the only dependence of these dynamics on the stimulus X is that the transition from state (3) to state (1) at time t occurs with probability $f(x_t)$. We observe only the spikes (or a noisy function thereof); the state variables are never observed directly. Aside from the choice of response function $f(x_t)$, the only free parameter is p_{23}, which determines the average length of the refractory period[1]. In all of our experiments (results shown below) we assume that the neuronal population is composed of conditionally independent neurons of exactly this type, with $f(x_t) = x_t$ or $f(x_t) = 1 - x_t$. This choice is for simplicity and is not essential, as will become clear below; some conditional interdependence among the neurons is possible with only a slight cost in tractability, for example, and the stimulus-dependence of the various transition probabilities could similarly be more complicated.

Next we describe the model assumptions corresponding to each of the information-loss

[1]Note that the refractory period here is relative. Markov models that impose an absolute refractory period are easy to imagine. Indeed, similar models have been considered that include an absolute refractory period via some number of additional refractory states that the neuron passes through deterministically (e.g., [Herbst et al., 2008; Haslinger et al., 2010]). Our methods apply to such models as well.

mechanisms discussed above.

3.4.1 Spike time jitter

In this setting we assume that we can only observe temporally-jittered versions of the true spike times. For concreteness, we assume that the jittered spike times have been drawn from discretized Gaussian distributions with standard deviation σ centered on the actual (unknown) spike time. We assume that σ is known or can be estimated based on previous experiments. (All of these assumptions can be relaxed considerably.)

3.4.2 Identity loss

In this setting we focus on spike-sorting errors. Observations here correspond to spike feature vectors, which represent, e.g., the amplitude and width of the spike voltage waveform, or a projection onto principal components in voltage waveform space. The distributions of these spike features overlap, so it is impossible to say with certainty which spike came from which neuron. For concreteness, we assume that these feature vectors are two-dimensional and drawn from Gaussian distributions whose mean and covariance are known or can be estimated. (Again, all of these assumptions can be relaxed considerably.)

3.5 Stimulus decoding

In both of the settings described above, the accuracy of stimulus reconstructions provides a measure of the information conveyed in the observed spike data. For a given stimulus prior $p(X)$ and noise distribution $p(\mathbf{Y}|X)$, we would like to, for example, compute the mean summed squared error (MSE),

$$\mathbb{E}_{P(X,\mathbf{Y})} \sum_t (x_t - \hat{x}_t)^2 , \tag{3.1}$$

where $\hat{X}$ is the posterior mean $\mathbb{E}(X|\mathbf{Y})$, the optimal stimulus reconstruction for the given $\mathbf{Y}$ under the square error loss function. We use bold-face $\mathbf{Y}$ to refer to the noisy signals from all neurons, and Y to refer to the noisy signals from just one neuron. Similarly, we use $\mathbf{Q}$ to refer to the state sequence of the assembly of neurons, and Q to refer to a single

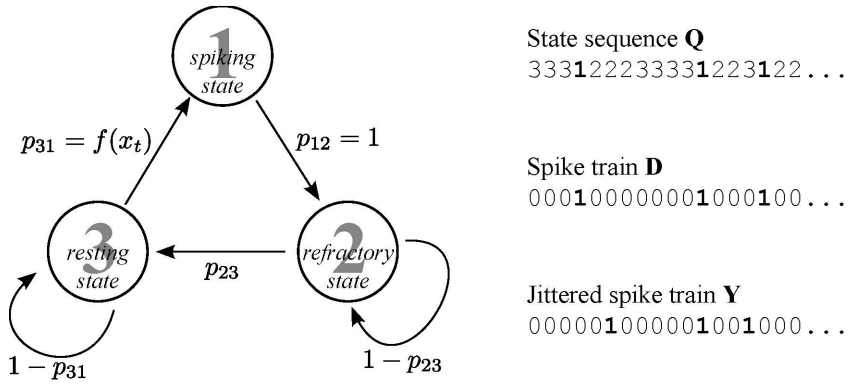


Figure 3.2: Left: A simple discrete-time model for a neuron responding to a time-varying stimulus $X = \{x_t\}$. A neuron in the resting state (3) enters the spiking state (1) with a probability that is a function of the current stimulus value. The neuron passes immediately from (1) to a refractory state (2) where it stays for a time drawn from the geometric distribution with parameter p_{23}, which is not a function of the stimulus. Right: example values of the hidden state sequence Q, the true (unobserved) spike times D, and the observed spike data Y.

neuron's state sequence. When appropriate, we include a superscript index to specify the neuron, as in Q^i.

Our approach to compute the expression in (3.1), is to 1) draw many stimuli X from the stimulus prior $P(X)$, 2) for each stimulus, draw a sample state sequence $\mathbf{Q}$ and noisy observations $\mathbf{Y}$ from the model $P(\mathbf{Q}, \mathbf{Y}|X)$, 3) decode each set of noisy observations separately to obtain $\hat{X}$, and finally 4) compute the squared error $\sum_t (x_t - \hat{x}_t)^2$ for each such sample and average to obtain an unbiased Monte Carlo estimate of the expected loss (3.1). Steps 1, 2 and 4 are straightforward. For these Markovian neuron models, step 3 may be accomplished by the use (described below) of Gibbs sampling [Robert and Casella, 2005] in time that scales linearly with the duration of the experiment, i.e. the number of observations.

3.5.1 Gibbs sampling and Rao-Blackwellizaition

Gibbs sampling is a Markov chain Monte Carlo (MCMC) algorithm for sampling from complicated joint distributions $p(Z) = p(z_1, z_2, \cdots, z_n)$ whose conditional distributions $p(z_i|Z\backslash z_i)$ are relatively easy to sample from, where $Z\backslash z_i$ is the set of components of Z

except for z_i [Robert and Casella, 2005]. The algorithm proceeds by first initializing all the components of Z with some typical values. Then one cycles through the components (in whatever order, so long as all components are iterated over), sampling each z_i from its conditional distribution $p(z_i|Z\backslash z_i)$, holding all the other components fixed at their previous values. Then the value of z_i is updated to the sampled value before moving on to the next component in the cycle, and that value may be added to a tally. To estimate the marginal means of the components, these tallies are finally divided by the overall number of cycles. Since Gibbs sampling provides samples from the full joint distribution $p(Z)$, we may use these samples to compute estimates of any desired marginal moments or quantiles.

In an important variant of Gibbs sampling which we employ in the stimulus decoding problem, for each cycle and for each component, the conditional mean of the component, rather than the sampled value of the component, is tallied in the estimation of the marginal mean of the component. When this is possible, i.e. when the conditional expectation $\mathbb{E}(z_i|Z\backslash z_i)$ is known for any value of the other components $Z\backslash z_i$, it can be shown that the variance of this estimator is less than that of the average of sample values. Such an algorithm is termed a Rao-Blackwellized Gibbs sampler [Robert and Casella, 2005]. More generally, when feasible, we can record the full conditional distribution $p(z_i|Z\backslash z_i)$ at each iteration, and therefore obtain better estimates of the conditional variance, quantiles, etc., as well.

The Gibbs approach is most useful when the sampling from these conditional distributions is simpler than sampling from the joint distribution directly, as is the case in the present stimulus decoding problem. The details of this application are detailed below.

3.5.2 The stimulus decoder Gibbs sampler

We proceed by conditioning on the hidden state sequence $\mathbf{Q}$ that led to the generated spike train D; the probability of a stimulus X given the observed data $\mathbf{Y}$ may be written as the marginal probability $p(X|\mathbf{Y}) = \int p(X, \mathbf{Q}|\mathbf{Y}) d\mathbf{Q}$. Thus, to obtain a sample from $p(X|\mathbf{Y})$, we draw samples from $p(X, \mathbf{Q}|\mathbf{Y})$ but only record the value of X. We sample from this joint distribution by Gibbs sampling: given the sample $\mathbf{Q}^{(i)}$, we draw $X^{(i)}$ from $p(X|\mathbf{Q}^{(i)}, \mathbf{Y})$ and record $X^{(i)}$ (or the Rao-Blackwellized statistics $p(X|\mathbf{Q}^{(i)}, \mathbf{Y})$ or $E(X|\mathbf{Q}^{(i)}, \mathbf{Y})$, if these are

analytically available), then draw $\mathbf{Q}^{(i+1)}$ from $p(\mathbf{Q}|X^{(i)}, \mathbf{Y})$, and iterate. Fig. 3.3 shows an example sequence of samples of X and $\mathbf{Q}$.

The problem of stimulus reconstruction has been reduced to the problem of sampling from these two conditional distributions, $p(X|\mathbf{Q}, \mathbf{Y})$ and $p(\mathbf{Q}|X, \mathbf{Y})$. What remains is to sample from these conditional distributions in an efficient way; in particular, since we are interested in reconstructing stimuli given long samples of spike train data, it is important to develop methods that scale linearly with the length of the observed spike train data. In the following we will address each subproblem in turn, and we will illustrate the procedure with the simple three-state Markov model neuron introduced above.

Both in the jitter and identity loss cases, sampling from $p(\mathbf{Q}|X, \mathbf{Y})$ will be a matter of framing the dynamics and noisy signal as a hidden Markov model. Once that is done, we can sample from $p(\mathbf{Q}|X, \mathbf{Y})$ using standard forward-backward recursions [Rabiner, 1989]. Sampling from $p(X|\mathbf{Q}, \mathbf{Y})$ turns out to require similar approaches in both jitter and identity loss cases, and will involve a simple application of Bayes rule to write down the distribution to be sampled from. In the case that this distribution can be integrated analytically, we can employ Rao-Blackwellization.

3.5.3 Hidden Markov models

Hidden Markov models (HMMs) [Rabiner, 1989] are convenient graphical models for the analysis of discrete-time systems. HMMs represent joint distributions over latent (unobserved) variables on which some observed variables depend. To be concrete, a HMM consists of two components. First is a discrete Markov chain of latent variables $Z = \{z_t\}_{t=1}^T$, so that

$$p(z_t|z_1, \cdots, z_{t-1}) = p(z_t|z_{t-1}).$$

Second is a discrete set of observations $W = \{w_t\}_{t=1}^T$, one observation corresponding to each latent variable, and each of which has the Markov property

$$p(w_t|Z, W \backslash w_t) = p(w_t|z_t).$$

Many systems can be reasonably modelled by HMMs. In our case, the variables w_t will correspond to the binary presence or absence of a spike at time t, or the observed spike feature

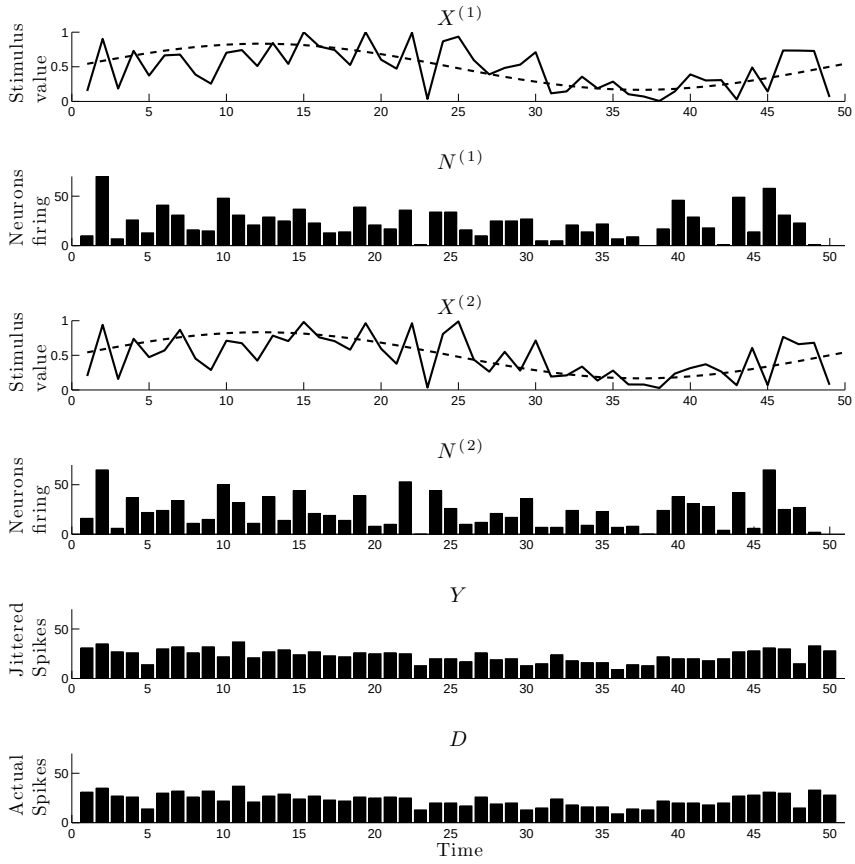


Figure 3.3: An example sequence of samples from $p(X|\mathbf{Q})$ and $p(\mathbf{Q}|X, \mathbf{Y})$, drawn via the Gibbs method described in the main text. The dashed curve is the actual stimulus. The solid curves in the stimulus panels are samples of X. The bars in the neurons firing panels show the number N of neurons firing at each time bin in the sample state sequence sample $\mathbf{Q}$. I.e., N_t is the number of neurons in the spike state ($q_t = 1$) at time t. These data are from a simulation of 300 neurons responding to a sinusoidal stimulus with $p_{23} = 0.1$. The jitter amplitude is $\sigma = 2$. 100 equilibration sweeps were made before these samples were recorded. The bottom two panels show the number of spikes in the jittered and actual spike trains at each point in time.

vector at time t; the z_t variables correspond to the unobserved neuronal state illustrated in Fig. 3.2.

It turns out that inference in a HMM – i.e. estimation of the latent variables given only values of the observed variables – is particularly efficient. In particular, inference time is linear in the length of the Markov chain. When Z and W form a hidden Markov model (HMM), sampling from the posterior state distribution $p(Z|W)$ proceeds via the standard filter-forward sample-backward HMM recursion [Fruhwirth-Schnatter, 2006]. We first compute the "forward probabilities"

$$a_t(i) = p(w_1, \cdots, w_t, z_t = i),$$

which can be computed recursively in $O(N^2T)$ time, where N is the number of possible states of each z_i. The forward variables are indexed by state, $i = 1, 2, \cdots, N$, and form a $N \times T$ matrix. We then recurse backwards to sample:

$$\begin{aligned} z_T &\sim p(z_T = i|W) = \frac{a_T(i)}{\sum_i a_T(i)} \\ z_t &\sim p(z_t|Z_{t+1:T}, W) \\ &= p(z_t|z_{t+1}, W) \\ &\propto p(z_{t+1}|z_t)p(z_t|W_{1:t}) \\ &\propto p(z_{t+1}|z_t)a_t(z_t). \end{aligned}$$

In the first line, the use of "$\sim$" means that z_T is drawn from the distribution $p(z_T = i|W)$. By appending all the sampled z_t variables into a single vector Z, we obtain a sample from the full conditional distribution $p(Z|W)$ in $O(T)$ total time.

The model parameters themselves (the transition probabilities $\alpha_{ij} = p(z_{t+1} = j|z_t = i)$ and the observation probabilities $\eta_{ij} = p(w_t = j|z_t = i)$ can be inferred, e.g., via the standard Expectation Maximization algorithm [Rabiner, 1989]; see [Escola et al., 2011] for further discussion. In the following we take the model parameters as known for simplicity.

3.5.4 Sampling from the posterior state distribution $p(Q|X, Y)$

To apply the efficient inference methods associated with HMMs to our information loss settings, we must frame each as a HMM but with the noisy (as opposed to the actual)

spikes as the observations. This entails characterizing the neurons in each setting as a system whose states $Q_{1:T}$ form a Markov chain, given stimulus X, and whose noisy observations $Y_{1:T}$ depend on the states $Q_{1:T}$ in a Markovian fashion [Rabiner, 1989]. In either setting, the neurons would trivially form a HMM if not for the information loss; for any Markovian neuron model, whether or not there is a spike at a given time bin depends only on the neuron's state q_t at that time. Once the information loss mechanism is introduced, however, it is the noisy signal $Y_{1:T}$ (i.e., the jittered spike times, or the observed noisy feature vectors) that we must formulate as the observation of a conditional HMM.

3.5.4.1 Spike time jitter

We seek to frame the generation of spike trains in the jitter setting as a HMM. However, note that each neuron's spike train and state sequence are conditionally independent of all other neurons' spike trains and state sequences, given the stimulus. Therefore, to sample the whole set $\mathbf{Q}$ of state sequences given all the jittered spike trains $\mathbf{Y}$ and given the stimulus X, we can sample Q given Y for each neuron in isolation.

Consider a neuron with Markovian dynamics as described above, so that D is its unjittered spike train, with spike times $\{t_i\}$, $i = 1, \cdots, N_{spikes}$. Then that neuron's state sequence Q forms a HMM with D as the observations. Given D and the stimulus X and with knowledge of the underlying Markovian dynamics of the neuron, we can sample $Q \sim p(Q|X, D)$ by the standard filter forward sample backward algorithm.

If instead we are given a jittered spike train Y, with jittered spike times $\{y_i\}$, $i = 1, \cdots, N_{spikes}$, then sampling Q is not so straightforward. Y and Q do not form a HMM. To see this, consider a given jittered spike time y_i. This spike corresponds to some true spike time t, whose spike is the observation from the neuron in state q_t. However, t could be anywhere in a window of time bins surrounding y_i. The width of this window is determined by the jitter amplitude. In this sense the spike time y_i may contain information regarding the state of the neuron at any of several or many time bins. We therefore need to take an alternate approach to sampling Q in the jittered case.

One approach is to break up the sampling into two Gibbs steps. First, sample the true spike train D, given X and Y. Then, with D in hand, sample $Q \sim p(Q|X, D)$ by the

forward backward recursion described above. (Note that $p(Q|X, D, Y) = p(Q|X, D)$, since Y is nothing but a corrupted copy of D.) There are many options for sampling D. One straightforward approach is to use Gibbs again, this time sampling one spike at a time. (Of course in some cases — for example, in the case of a highly bursty neuron — it might make more sense to move multiple spikes at once. The discussion below can easily be generalized to this case; see, e.g., [Mishchenko and Paninski, 2011] for further discussion.) More precisely, for each $1 \leq i \leq N_{spikes}$, we sample t_i from

$$p(t_i = t|\{t_{j\neq i}\}, X, Y) \propto p(Y|t_i = t, \{t_{j\neq i}\})p(t_i = t|\{t_{j\neq i}\}, X). \tag{3.2}$$

The second factor on the right hand side can be computed by a forward-backward recursion. (It is worth noting that we do not have to perform the full recursion every time we update a spike time t_i: a local change in t_i will have only a local effect on the values of the forward and backward probabilities. As we recursively update these probabilities, therefore, we can stop once we see that the difference between the updated values and the previous values is negligible. This ensures that the cost of a spike time update does not scale with the total length of the spike train.)

To compute the the first factor on the right hand side of eq. (3.2), we need to specify a model for the temporal noise process that defines $p(Y|D)$. The simplest model is that each spike time is jittered independently:

$$p(Y|D) = \sum_{\sigma} \prod_{j=1}^{N_{spikes}} p(y_{\sigma(j)}|t_j).$$

The sum over σ here is over all $N_{spikes}!$ possible permutations (relabelings) of the spikes, accounting for the fact that we don't know which spike in the observed set Y corresponds to a given spike t_i in the unobserved true set D. However, once we pick a labeling σ that maps D to Y, then computing the conditional probability of Y just reduces to a product over the individual jitter densities $p(y_{\sigma(j)}|t_j)$.

Clearly, direct computation of the sum over σ becomes intractable as N_{spikes} becomes large, since the number of terms in the sum grows exponentially. However, note that we do not have to compute the full sum. Instead, we just need to compute the change in the sum as the i-th spike is moved from the current time t_i to a new time t_i'. If the variance of

the jitter distributions $p(y_j|t_j)$ is not too large, any given spike time t_j will only be close to a few observed spikes y_i, and the computation of the change in $p(Y|D)$ becomes tractable. As a concrete example, if y_i is sufficiently distant from the nearest other observed spikes that we can neglect the probability that the spike times have been switched by the jitter, then we can uniquely associate the observed spike y_i with a single true unobserved spike t_i, and we have

$$\begin{aligned}\frac{p(Y|D)}{p(Y|D')} &= \frac{\sum_\sigma p(y_{\sigma(i)}|t_i)\prod_{j\neq i} p(y_{\sigma(j)}|t_j)}{\sum_\sigma p(y_{\sigma(i)}|t'_i)\prod_{j\neq i} p(y_{\sigma(j)}|t_j)} \\ &\approx \frac{p(y_i|t_i)\sum_\sigma \prod_{j\neq i} p(y_{\sigma(j)}|t_j)}{p(y_i|t'_i)\sum_\sigma \prod_{j\neq i} p(y_{\sigma(j)}|t_j)} \\ &= \frac{p(y_i|t_i)}{p(y_i|t'_i)}.\end{aligned}$$

(We have abused notation slightly here: this sum over σ includes all permutations of the $N_{spikes} - 1$ spike times not including t_i or y_i.) Thus we can now combine the two necessary factors in eq. (3.2), and update t_i either via standard Gibbs or a Metropolis-within-Gibbs [Robert and Casella, 2005] approach. (See [Chen et al., 2009; Tokdar et al., 2010] for some related approaches.)

3.5.4.2 Neural identity loss; spike addition or deletion

In the identity loss / spike sorting setting, any given observed spike could have been emitted from any of multiple neurons. Separate independent neuron models cannot capture such ambiguity, and therefore we must combine the models across neurons appropriately.

One direct approach is to construct a single state space that represents the dynamics of all the neurons at a given recording electrode. Suppose there are m discernible neurons at a given location, each with at most K Markovian internal states. We can track the state of all of these neurons simultaneously by forming the direct product of each individual state's transition and observation matrices (obtaining a joint state variable that can take on at most K^m possible values), as discussed in [Calabrese and Paninski, 2011]. If each neuron i has a transition matrix P_i, then the joint state transition matrix is the Kronecker product [Horn, 1986] of the single-neuron transition matrices,

$$P = \bigotimes_{i=1}^{m} P_i.$$

To construct a HMM it remains to specify an emission matrix, whose entry (i, j) is the probability of seeing observation i given that the system is in state j. For our three state neuron, observed without noise, this would be a 3×2 matrix:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix},$$

where the rows correspond, from top to bottom, to the states *firing*, *refractory*, and *resting*, and the columns correspond, from left to right, to the observations *firing* and *not firing*. Changing this emission matrix would be one way of modeling spontaneous spiking or dropped spikes.

In the spike sorting setting, we replace this simple discrete observation alphabet with a continuous, typically multi-dimensional feature vector $\mathbf{y}$ defined in terms of, e.g., the amplitude of the voltage (or the magnitude of some projection onto a principal component) triggered on a threshold crossing. We write this in terms of the density function

$$b_{\mathbf{q}}(\mathbf{y}) = \text{Prob}[\mathbf{y}_t = \mathbf{y} | \mathbf{q}_t = \mathbf{q}].$$

If we assume for concreteness that the observed feature vectors have a Gaussian distribution given the identity of the neuron that emitted the spike, then $b_{\mathbf{q}}(\mathbf{y})$ is readily evaluated as the Gaussian density at $\mathbf{y}$, with mean and covariance corresponding to the neuron that is firing (as indexed by the multineuron state $\mathbf{q}$).

We can easily extend this model to treat the possibility of dropped spikes or spontaneous spiking. To account for dropped spikes, we add a cluster corresponding to no spike, with its own mean and covariance. Assume that the probability of a spike being dropped is p_d. Then whenever $\mathbf{q}$ is a spiking state for one of the neurons, there is a chance, p_d, that the observation is drawn from the cluster corresponding to the absence of a spike. The observation density may be modified accordingly to handle this possibility. See also [Goodman and Johnson, 2008], who find that in at least some cases false positives will have a greater negative effect on decoding than either dropped or mislabeled spikes.

If all the neurons at some electrode have the same response function (a very special case), then all we are interested in is the number of neurons in each state. In this case the state

space size scales polynomially in the number of neurons m at the recording location, not exponentially. The transition probabilities in this condensed state-space may be computed as follows. Define the vector $\vec{n}_t$ such that $n_t^{(k)}$ is the number of neurons in state k at time t. For each state k, define the vector $\vec{\nu}_{tk}$ as follows: $\nu_{tk}^{(i)}$ is the number of neurons in state i at time t that were in state k at time $t-1$. Then $\vec{n}_t = \sum_k \vec{\nu}_{tk}$. The transitions of the neurons in state i at time $t-1$ are independent of the transitions of the neurons in state $j \neq i$ at times $t-1$. Therefore

$$\begin{aligned} p(\vec{n}_t|\vec{n}_{t-1}) &= p\left(\sum_k \vec{\nu}_{tk} \middle| \vec{n}_{t-1}\right) \\ &= conv\left[p\left(\vec{\nu}_{tk}\left|n_{t-1}^{(k)}\right.\right)\right] \\ &= conv\left[Mult\left(n_{t-1}^{(k)}, p(q_t|q_{t-1}=k)\right)\right] \end{aligned}$$

where the q_t index the single-neuron states, $conv()$ denotes m-fold discrete convolution, and $Mult(N,p)$ denotes the multinomial distribution with parameters N and p. The first equality follows by definition. The second equality follows because $\vec{n}_t$ is the sum of independent random variables $\vec{\nu}_{tk}$, and because $p\left(\vec{\nu}_{tk}\,|\vec{n}_{t-1}\right) = p\left(\vec{\nu}_{tk}\left|n_{t-1}^{(k)}\right.\right)$, i.e. $\vec{\nu}_{tk}$ is conditionally independent of the other components of $\vec{n}_{t-1}$. The transition of each neuron from state k to some state i is a trial with a fixed finite number of possible outcomes, independent of all other trials. Therefore each vector $\vec{\nu}_{tk}$ is drawn from a multinomial distribution with $n_{t-1}^{(k)}$ trials and event probabilities $p(q_t = i|q_{t-1}), i = 1, \cdots, K$.

Finally, note that we can easily extend our model to treat multiple spikes in a time bin, whether by the same neuron or by different neurons at the same electrode, but the model becomes more complex as the time bin width increases, and as the maximum possible number of spikes per bin grows. For example, at an electrode monitoring two neurons, A and B, multiple spikes could be treated by the addition of Gaussian modes for events like "A spikes, B does not", "A does not spike, B spikes twice", "A and B both spike once", and so on.

In summary, whether or not all neurons have the same transition matrix, we have established that we may sample from the posterior $p(\mathbf{Q}|X,\mathbf{Y})$ in a computationally efficient way. However, the computational cost does grow exponentially with the number of neurons

m in a single cluster. If m is very large, it becomes more attractive to use a blockwise-Gibbs approach (as described, e.g., in [Mishchenko and Paninski, 2011]), in which we sample the states of a small subset of neurons while holding the states of the other neurons fixed.

3.5.5 Sampling from the posterior stimulus distribution $p(X|Q,Y)$

We have shown that we can tractably sample from one side of the Gibbs sampler decoder (the $p(\mathbf{Q}|X,\mathbf{Y})$ side) in a way that scales linearly with the temporal length T of the dataset to be decoded. In the other stage, we sample the stimulus X from $p(X|\mathbf{Q},\mathbf{Y})$. We will show that this half of the Gibbs sampler is relatively straightfoward; moreover, this step can be made particularly efficient in certain special cases.

First note that $\mathbf{Y}$ contains no more information about the stimulus than $\mathbf{Q}$, so that $p(X|\mathbf{Q},\mathbf{Y}) = p(X|\mathbf{Q}) \propto p(\mathbf{Q}|X)p(X)$. The conditional density $p(\mathbf{Q}|X)$ factorizes into a product of its transition probabilities: for each neuron,

$$p(Q|X) \propto p(q_0)\prod_t p(q_{t+1}|q_t, x_t),$$

by the Markov assumption.

If we restrict our model to log-concave transition probabilities $p(q_{t+1}|q_t, x_t)$ (i.e., assume that $\log p(q_{t+1}|q_t, x_t)$ is concave in x_t), and if our stimulus prior is log-concave in X as well, then the posterior on X is log-concave and therefore unimodal, which means that the sampler will not get trapped in local optima, and we can use efficient sampling methods as discussed in [Ahmadian et al., 2011]. The restriction to log-concave nonlinearities is not severe in our class of Markov models; see e.g. [Escola et al., 2011] for further discussion.

If we specialize further we can obtain an even more efficient sampler. In our simple three-state model, let's assume that the transitions $p(q_{t+1} = \text{“}spike\text{”}|q_t = \text{“}rest\text{”}, x_t) = f(x_t)$ depend on x_t in a simple linear fashion: either $f(x_t) = x_t$ (for cells of “ON” type) or $f(x_t) = 1 - x_t$ (for cells of “OFF” type). In this case the likelihood term $p(\mathbf{Q}|X)$ can be written in a very simple form:

$$p(X|\mathbf{Q},\mathbf{Y}) \propto \prod_t x_t^{C(t)}(1-x_t)^{D(t)} \tag{3.3}$$

where $C(t)$ counts the number of ON neurons passing into the spike state from the rest

state, plus the number of OFF neurons remaining in the rest state at time t; $D(t)$ is defined similarly, but with ON and OFF reversed.

This likelihood term leads to a remarkably simple posterior if $p(X)$ has a similar form. For example, if x_t is chosen independently at each time step, with a uniform distribution on the interval $[0, 1]$, then the posterior on X is given by a simple independent product of beta distributions, which can be sampled trivially. (Similar expressions involving sums of incomplete beta functions hold for the case that the stimulus prior is composed of an independent product of polynomials, or if the response functions $f(.)$ have a more general polynomial form.)

Of course, the assumption that each x_t is a priori independent in time is typically too strong. In general, the choice of stimulus prior will depend on the details of the particular experimental setup. What is most important is to have available a broad class of tractable models to choose from. [Smith et al., 2012] recently introduced a class of models which we can use to provide a convenient correlated prior for X. These so-called low-rank models are joint distributions whose structure allows for fast exact inference in settings where standard models such as the discrete HMM or linear Gaussian models are not applicable. These models consist of joint distributions over continuous variables X whose dependency structure can be expressed via discrete latent variables $Z = \{z_t\}$ coupling the main variables X. In the case that $p(X)$ forms a Markov chain, such a low-rank $p(X)$ may be decomposed as follows:

$$\begin{aligned} p(X) &= p(x_1) \prod_{t=1}^{T-1} p(x_{t+1}|x_t) \\ &= p(x_1) \prod_{t=1}^{T-1} \sum_{z_t}^{R} p(z_t|x_t) p(x_{t+1}|z_t) \end{aligned} \tag{3.4}$$

for appropriate discrete auxiliary variables z_t. (The first equation above is the standard Markov condition; the second equation expresses the low-rank nature of the conditionals $p(x_{t+1}|x_t)$, with R denoting the "rank" of the model.) See [Smith et al., 2012] for full details; the key point is that exact inference over X is tractable in this case via standard forward-backward recursions in $O(TR^2)$ time, despite the fact that the x_t variables may be non-discrete and highly non-Gaussian. As one useful example of a low-rank model, consider

$p(X)$ of the nearest-neighbor polynomial form

$$p(X) \propto \prod_t \sum_{i=0}^{R} a_{ti} x_t^{\alpha_i} (1-x_t)^{\beta_i} x_{t+1}^{\gamma_i} (1-x_{t+1})^{\delta_i},\ 0 \le x_t \le 1. \tag{3.5}$$

(The normalization constant of $p(X)$ is found via the same forward recursion as is used to perform inference.) As discussed in [Smith et al., 2012], this class of priors has several helpful features. First, with an appropriate choice of the polynomial order R and the coefficients $\{a_{ti}\}$, we have a good deal of flexibility in modeling the correlations in x_t. Second, this prior is conjugate to the likelihood term (3.3); i.e., the posterior has the same form as the prior. Finally, by the general theory mentioned above, exact samples can be drawn from this prior (and posterior) in $O(TR^2)$ time. As an example, Fig. 3.4 illustrates the effect of changing the prior when computing the posterior expectation $E(X|\mathbf{Q})$. In particular, we used

$$p(X) \propto \prod_t \sum_{i=0}^{R} \binom{R}{i}^2 x_t^i (1-x_t)^{R-i} x_{t+1}^i (1-x_{t+1})^{R-i} \tag{3.6}$$

for two different choices of the parameter R, which (as discussed further in [Smith et al., 2012]) serves to set the smoothness of samples from $p(X)$: larger values of R correspond to smoother samples from the prior (and in turn to a smoother posterior expectation). (We point out that, as discussed in [Smith et al., 2012], R can be chosen by standard model-selection methods (e.g., maximum marginal likelihood); in addition, for sources of signals that tend to have occasional large jumps, we can add a "slab" term – i.e. a small constant – to each pair potential to allow for such jumps, and that such a prior would still be an instance of Eq. (3.5).) We do not present detailed error statistics here, as this example is meant only to illustrate the behavior of the smoother given fully-observed spike trains; we will discuss applications to corrupted spike trains below.

We can further improve the efficiency of the sampler by the Rao-Blackwellization procedure discussed above, because it is also possible to compute the posterior moments $E(x_t|\mathbf{Q})$, $E(x_t^a|\mathbf{Q})$, $E(x_t x_s|\mathbf{Q})$, etc., using a similar $O(TR^2)$ forward-backward approach. Thus, for example, if we want to estimate the posterior mean $E(X|\mathbf{Y})$, instead of recording $X^{(i)}$ at each iteration of the Gibbs sampler, we record $E(X^{(i)}|\mathbf{Q}^{(i)})$, and average over these quantities at the end of the sampling run. In the more general case of log-concave posteriors

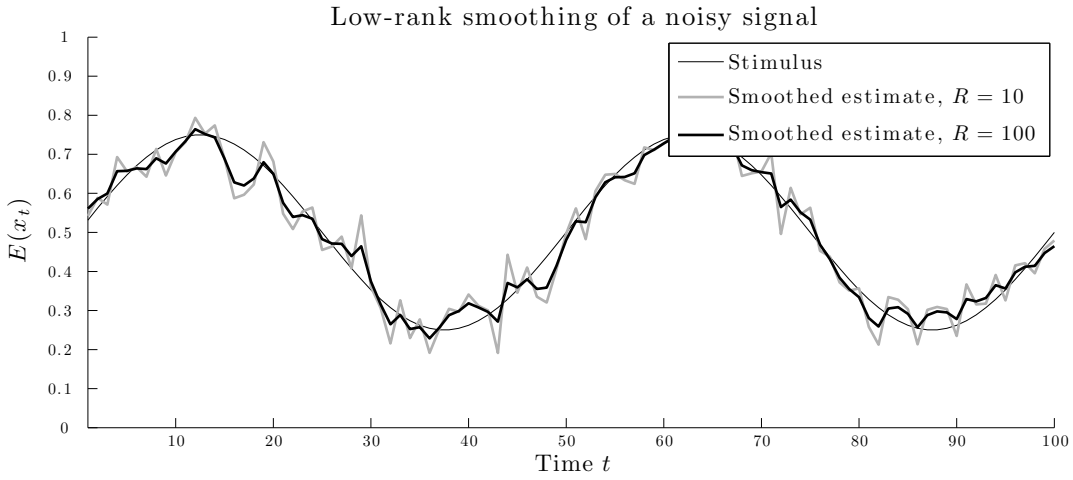


Figure 3.4: Estimating a stimulus from spike trains with a smoothing low-rank prior. A sinusoidal stimulus (thin black line) was the success probability for a neuron to fire, i.e. $p(n_t = 1|x_t) = x_t$, $n_t \in \{0, 1\}$; n_t not shown. The stimulus was estimated as its posterior mean $E(x_t|\{n_t\})$ with a low-rank smoothing prior of the form of Eq. (3.6) with ranks $R = 10$ and $R = 100$. Note that the spikes n_t were completely and noiselessly observed here; no spike train corruption has been applied.

on X, we typically can not Rao-Blackwellize exactly, but nonetheless we can often substitute an approximation to the mean to obtain a more efficient estimator. For example, for smooth and unimodal posteriors, the MAP estimate $\arg\max_X p(X|\mathbf{Q})$ tends to be a good approximation to the posterior mean $E(X|\mathbf{Q})$. [Pillow et al., 2011] discuss efficient methods for computing the MAP estimate, which can subsequently be plugged in to approximately Rao-Blackwellize the estimate of the posterior mean.

3.6 Results

3.6.1 Spike time jitter

Fig. 3.5 illustrates the effects of spike-time jitter on decoding accuracy of a stimulus drawn from a low-rank prior. We simulated the responses of 300 neurons driven by this stimulus, using the simple Markovian model described in Fig. 3.2, with $p_{23} = 0.1$ (recall that this parameter is inverse to the average refractory time) and firing rate $p_{31} = x_t$. The elements

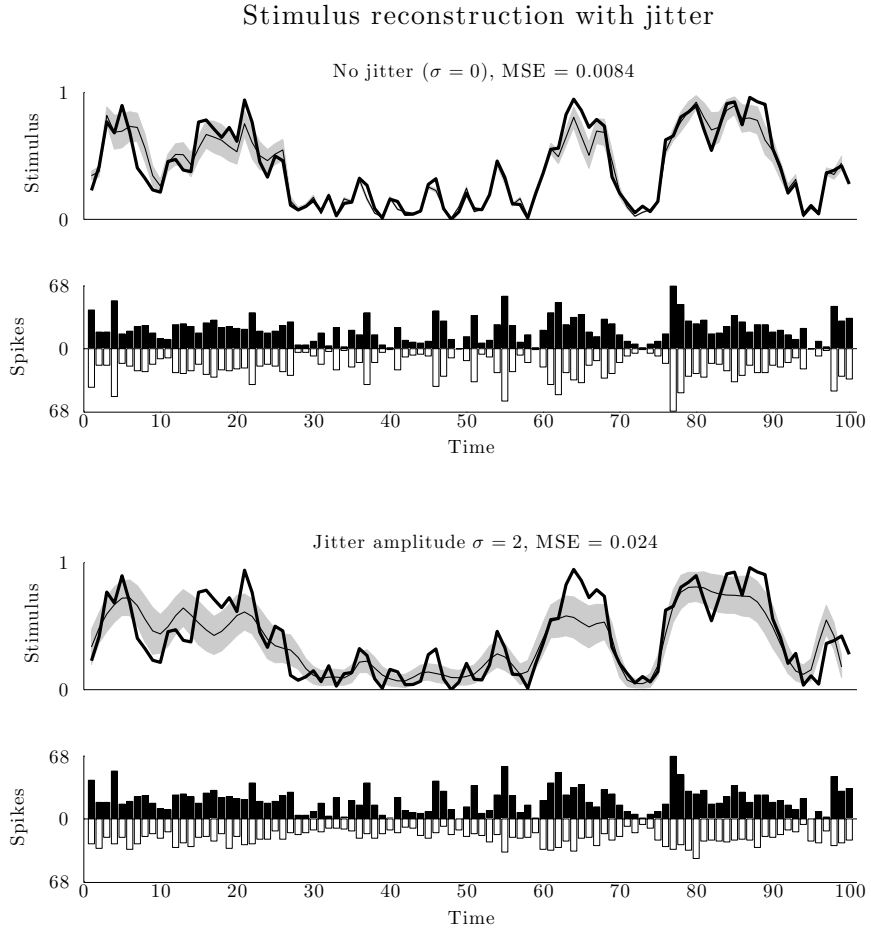


Figure 3.5: Effects of jitter scale on decoding accuracy for stimulus drawn from the prior of Eq. (3.6) with $R = 10$. In the decoder, the same low-rank prior was used. The reconstruction algorithm was run for 1,000 sweeps after 100 burn-in sweeps using 300 neurons and $p_{23} = 0.1$. The thick solid line is the actual stimulus. The thin solid line is the reconstruction, with plus or minus one posterior standard deviation shaded gray. The black and white bars show the number of actual (black) and jittered (white) spikes at each instant. The accuracy deteriorates with increasing jitter while the variance of the posterior density increases.

of the initial state probability vector were chosen at random. We then independently jittered each true spike according to a discretized Gaussian distribution with zero mean and

standard deviation 0 (top) and 2 (bottom). For these parameter values, the probability of spike crossing or overlap was negligible (recall the discussion in section 3.5.4.1 about the computation of $p(Y|Q)$). The Rao-Blackwellized Gibbs sampler was run for 1,000 iterations after a burn-in period of 100 sweeps, using a prior for X which was independent (stimulus values at different time bins are independent) and uniform on $[0, 1]$. As expected, the accuracy of the reconstruction diminishes with increasing jitter scale.

Fig. 3.6 summarizes how the reconstruction accuracy varies as the number of neurons changes, and also as the jitter amplitude grows, for this square-wave stimulus. Similar reconstructions for several different types of stimulus are shown in Fig. 3.7 and Fig. 3.8. In Fig. 3.7, the same independent uniform prior is used in the decoder. In Fig. 3.8, the decoder uses a low-rank prior of the form of Eq. (3.6). This corresponds to Eq. (3.4) where

$$x_1 \sim Uniform([0, 1])$$
$$z_t|x_t \sim Binomial(z_t; R, x_t)$$
$$x_{t+1}|z_t \sim Beta(z_t, R - z_t)$$

This simple beta-binomial form of the prior ($p(X)$) makes it possible to analytically compute the necessary integrals in the Rao-Blackwellized estimator.

The relative accuracy of the reconstruction of the jittered signal varies significantly across different test stimuli. This is consistent with the fact that none of these stimuli (except the one drawn from the independent uniform prior) are "typical" stimuli, under this stimulus prior. However, it is true for any sensible prior that the relative accuracy of the reconstruction will increase with the smoothness of the actual stimulus; the smoother the signal, the less information is lost by jittering a spike a small amount.

We have confirmed in our computer experiments that the computing time involved in inference in the spike time jitter setting scales linearly with the number of neurons as well as with the time T. For example, with 50 neurons, $p_{23} = 0.1$, jitter amplitude $\sigma = 1$, $T = 50$ time bins, and 1,000 Gibbs sweeps after 100 burn-in sweeps, estimation of a square-wave X took 41.7 seconds on a MacBook Pro with a 2.6 GHz Intel Core i7 processor and 8 GB 1600 MHz DDR3 RAM. For the same parameter values except for 100 neurons, estimation took 80.9 seconds. For the same parameter values except for 50 neurons and $T = 100$ time

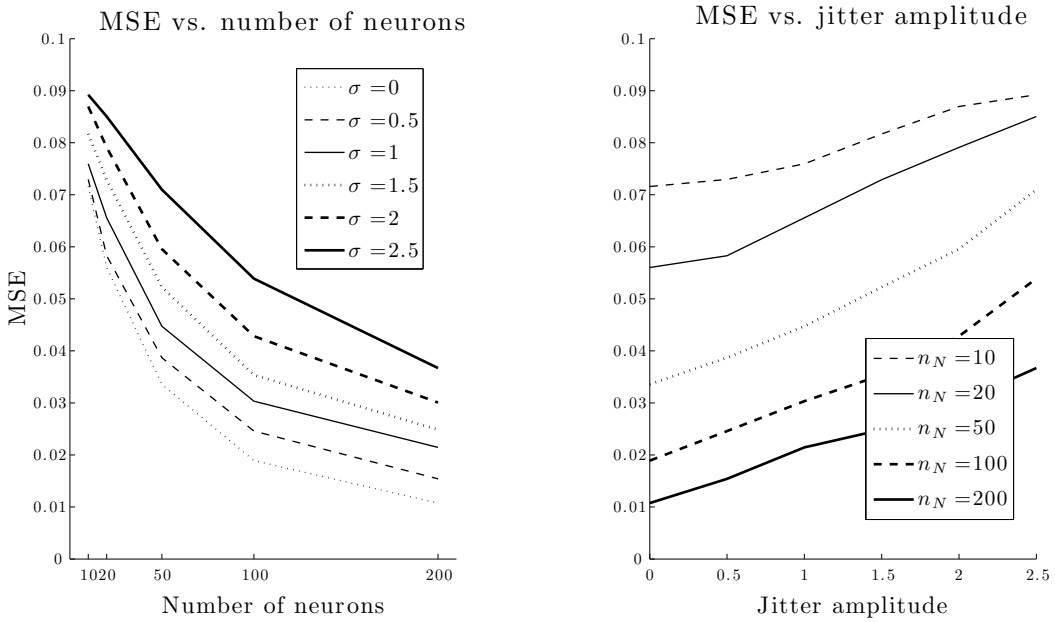


Figure 3.6: Accuracy (MSE) of reconstruction in the jitter case for various values of number of neurons and jitter amplitude, averaged over 100 reconstructions. A value of $p_{23} = 0.1$ was used for the refractory transition probability. The stimulus was modeled with a low-rank prior of the form of Eq. (3.6) with a rank of $R = 10$. The experiments ran for 1,000 Gibbs sweeps after a burn-in period of 100 sweeps. The actual stimulus used was a square wave with a period of 10 time bins.

bins, estimation took 79.5 seconds.

3.6.2 Identity loss

In the case of identity loss, we compare the performance of our decoder to that of two simple decoders. The first decoder is given the spike feature vectors and spike-times that our full decoder receives, computes the overall most likely state path for $\mathbf{Q}$ (i.e., the Viterbi path [Rabiner, 1989]) given the spike-times, and takes the spike-neuron assignments of this state path to be the true assignments; i.e., all uncertainty about the spike sorting is discarded. Subsequently, this decoder computes the posterior mean assuming these spike-neuron assignments – $E(X|D_{MAP}) = E(X|\arg\max_D p(D|Y))$. The second simple decoder

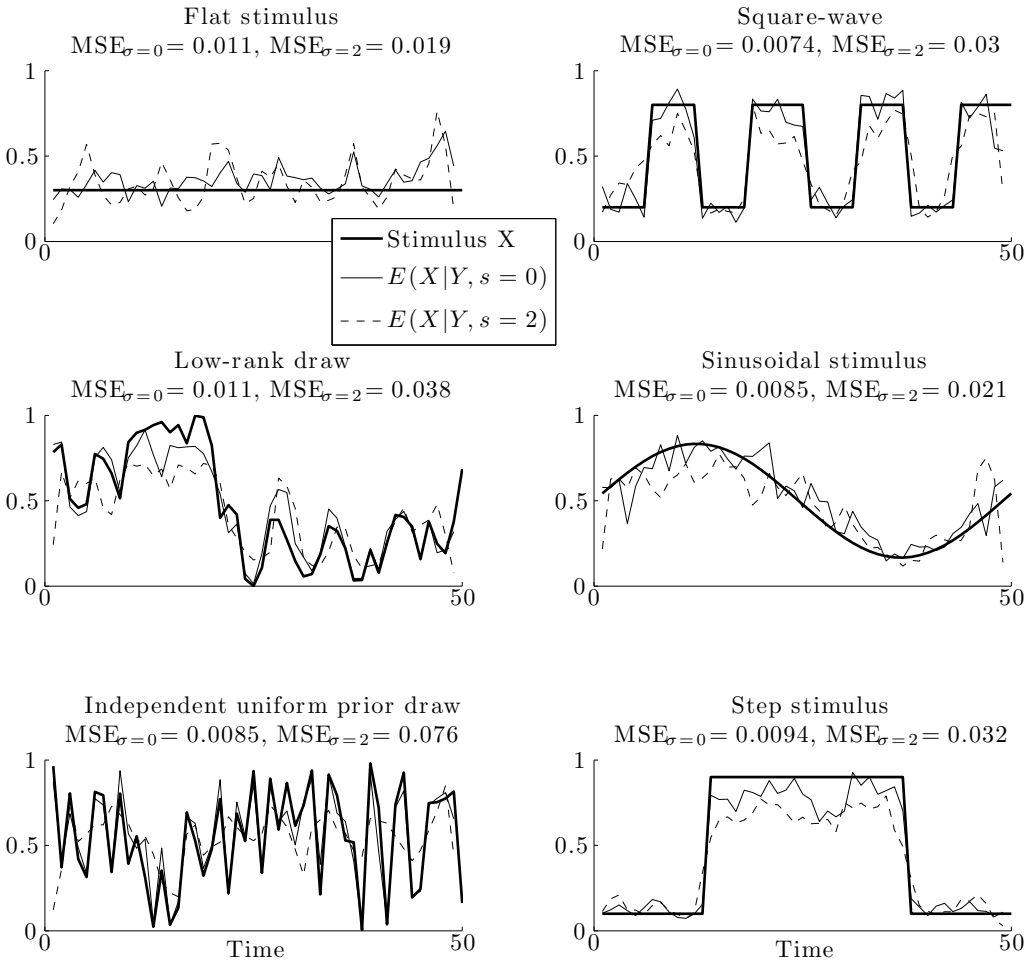


Figure 3.7: Reconstructions of different stimuli both for no jitter and for jitter using the independent uniform prior. The same parameter values as in Fig. 3.5 were used, except that 10,000 Gibbs sweeps were run after 1,000 burn-in sweeps. For very smooth stimuli (flat; sinusoidal) reconstruction from the jittered spike train may in fact be more accurate (in terms of MSE) than reconstruction from the original spike train. For less smooth stimuli (more typical of samples from the independent uniform prior) reconstruction from the original spike train is more accurate than reconstruction from the jittered spike train.

is given the actual spike-neuron assignments and subsequently computes the posterior mean $E(X|D)$. (We expect this decoder to outperform any decoder that does not have access

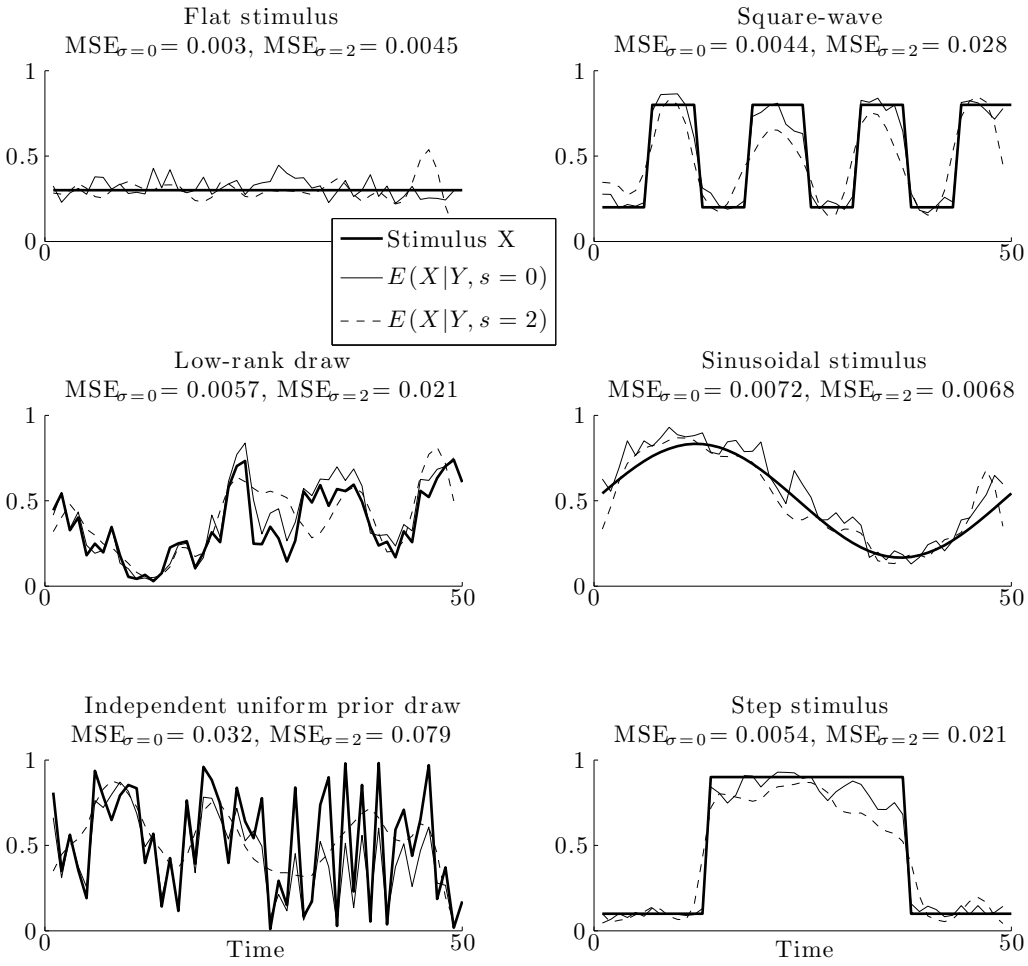


Figure 3.8: Reconstructions of different stimuli both for no jitter and for jitter using the low-rank prior of Eq. (3.6) with rank $R = 10$. The same parameter values as in Fig. 3.5 were used, except that 10,000 Gibbs sweeps were run after 1,000 burn-in sweeps. Here, smoother signals are more accurately estimated than they were with the independent uniform prior (see Fig. 3.7).

to the uncorrupted spike train data D.) In each case, in our experiments, an independent and uniform prior on X was used for the decoding. For each neuron, p_{23} was set at 0.5. We simulated 5-10 electrodes (see figure captions for specific values), on each of which two neurons were recorded. At each electrode, one of the neurons had the same transition matrix

as the simple Markovian model above. The other neuron had a transition matrix identical except that $p_{31} = 1 - x_t$ and $p_{33} = x_t$ instead of $p_{31} = x_t$ and $p_{33} = 1 - x_t$. (I.e., on each electrode we simulated a neuron of ON and OFF type.) The observed feature vectors were sampled for each neuron from a Gaussian distribution with each neuron's assigned mean and covariance. Reconstruction proceeded by Rao-Blackwellized Gibbs sampling, assuming zero jitter.

A sample stimulus along with its reconstructions for varying degrees of spike cluster overlap is shown in Fig. 3.9. Reconstruction quality is poor when the overlap between the feature Gaussian distributions is high (i.e., when spike sorting is challenging). The Bayesian estimate, however, stays closer to the mean, whereas the MAP-based estimate ranges widely, having committed to certain spike assignments to neurons. The same is true in the case of partial spike cluster overlap, where the reconstructions look somewhat better. The Bayesian decoder remains somewhat agnostic as to the assignment of spikes to neurons: the assignments vary from sweep to sweep of the sampler (data not shown). Meanwhile, the MAP-based decoder does not vary its assignments; since every sweep of the sampler will involve the same assignments, the reconstruction will be further from the mean than for the Bayesian decoder, with a smaller posterior variance, making this estimator "overconfident" and more frequently wrong than the fully Bayesian estimator.

We averaged the results of 100 reconstructions to compare the MSE of each decoder. Results are shown in Fig. 3.10. We find that that the full Bayesian decoder outperforms the Viterbi-spikes decoder. This practical example (Fig. 3.10) illustrates that properly accounting for spike sorting uncertainty can lead to significantly improved decoding, echoing results from earlier work (e.g., [Wood and Black, 2008; Ventura, 2008b; Ventura, 2008a; Chen et al., 2012]) using simpler, Poisson-based encoding models.

3.7 Conclusions and extensions

We have described methods for optimal Bayesian decoding of noisy or corrupted spike train data, and for quantification of the information lost due to several different possible sources of noise, including uncertainty in neural identity, spike deletion or insertion, and loss of tem-

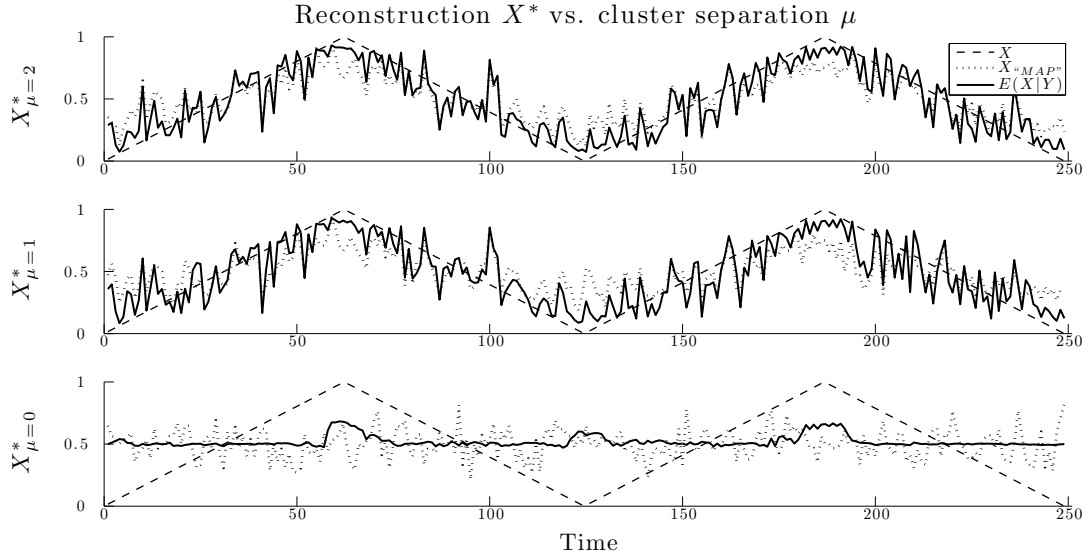


Figure 3.9: Sample reconstructions of a simple sawtooth stimulus for three examples of monotonically decreasing spike sorting difficulty. When spike clusters overlap completely (bottom panel), neither the MAP-based estimate nor the Bayesian estimate performs well. The Bayesian estimate, however, is more agnostic, providing estimates which are close to the posterior mean stimulus value. The MAP-based estimate "overcommits" to its assignments of spikes to neurons. In the case of partial overlap (middle panel), both the MAP-based and Bayesian estimates perform better, with the Bayesian estimate noticeably more accurate than the MAP-based estimate in several regions. In the case of negligible overlap (top panel), the three decoders produce nearly identical reconstructions. Reconstructions were conducted using 20 neurons (10 electrodes) and a uniform stimulus prior. The Gibbs sampler ran for 10,000 sweeps after a burn-in period of 1,000 sweeps.

poral resolution due to noise or low-temporal-resolution recording techniques. Our methods allow us to quantify the loss of decoding accuracy as a function of various biophysical and experimental variables of interest, including the jitter amplitude, stimulus frequency content (cf. [Goldwyn et al., 2010]), the number of neurons, firing rates, the refractoriness of the observed neurons, and so on. One practical example (Fig. 3.10) illustrates that properly accounting for spike sorting uncertainty can lead to significantly improved decoding, echoing results from earlier work (e.g., [Wood and Black, 2008; Ventura, 2008b; Ventura,

2008a; Chen et al., 2012]) using simpler, Poisson-based encoding models.

A couple of directions for potential future work are clear. First, we have focused on the case of neural populations which fire in a conditionally independent manner given the stimulus. Such an assumption might be sensible in the analysis of spike train data obtained via extracellular recordings where electrodes are relatively widely spaced. However, we have made this assumption here purely for the sake of clarity and simplicity. A good deal of recent work has focused on models which can account for additional dependence structure; see, e.g., [Vidne et al., 2012] for a recent review of this literature. It would be natural to extend our framework to handle models of this type, perhaps via the efficient blockwise-Gibbs samplers discussed in [Mishchenko and Paninski, 2011]. Second, while we have focused on a model-based approach here, there is a long history of more nonparametric jitter-based approaches for addressing hypotheses about the importance of temporal precision in the nervous system; see [Amarasingham et al., 2012] for a nice recent review. It would be interesting and valuable to explore further links between these nonparametric approaches and the parametric, decoding-oriented approach we have taken here. Finally, [Naud and Gerstner, 2012] consider the problem of stimulus decoding given observations of summed spike counts from a population of identical neurons. This can be considered a special case of our spike identity loss setting. It would be interesting to investigate the possibility of combining their analytical approaches with our MCMC-based techniques developed here.

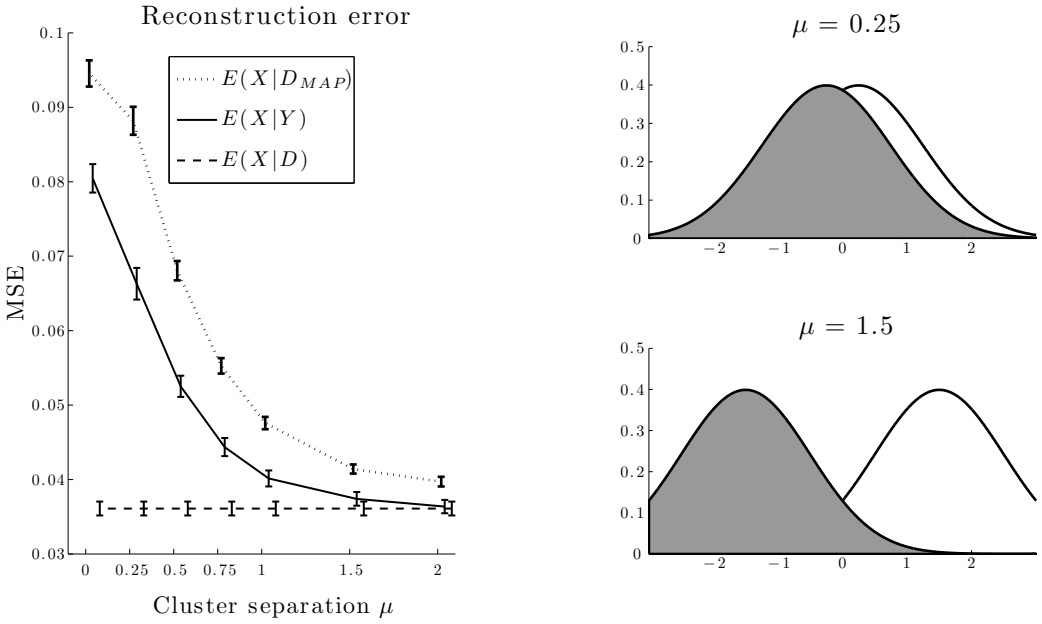


Figure 3.10: **Left**. Reconstruction error for the three spike-sorting decoders described in the main text. For each separation distance of the two spike clusters 100 reconstructions were performed of stimuli randomly drawn from the uniform prior over $[0, 1]^T$ incident on 10 neurons (5 electrodes) with refractory probability $p_{23} = 0.5$. The dotted line traces the error of the MAP-based estimate. The solid line traces that of the Bayes estimate, which consistently outperforms the MAP-based decoder. For reference, the performance of an optimal decoder that knows the correct spike assignments is included (dashed line). Samplers were run for 10,000 sweeps after burn-in periods of 1,000 sweeps. **Right**. Illustration of the degree of overlap in one-dimensional Gaussians with with unity standard deviation.

Chapter 4

Bayesian inference of sparse synaptic connectivity with spike-and-slab models

4.1 Abstract

We address the question of how to infer the locations of synapses onto dendritic trees of neurons. We assume a compartmentalized neuron and a dynamical model in which the time-varying voltages in all compartments are governed by a constant connectivity matrix (or weight matrix) which acts on the vector of inputs to the compartments. The experimenter only has direct access to a vector of noisy observations. *A priori* we assume that the true connectivity matrix is sparse, since presumably most compartments will not be very close to a synapse. Previous work [Pakman et al., 2012] applied and generalized the ideas of least angle regression [Efron et al., 2004a] to obtain a fast Bayesian solution to this estimation problem that imposes sparsity in the estimates of the weight matrix. A key feature of the dynamical model considered in that work is that the logarithm of the likelihood (probability of the data given the synpatic weights) is quadratic in the synaptic weights. In this work we explore two other approaches to the same problem. Both approaches assume the same likelihood. Where they differ from one another and from the model in [Pakman et al., 2012]

is the choice of prior over the synaptic weights. One approach employs the horseshoe model of [Carvalho et al., 2010], and the other uses spike-and-slab models.

4.2 Introduction

Understanding the synaptic organization of local neural circuits remains a central challenge in neuroscience.[1] To make progress towards this aim it would be of great value to measure the full synaptic connectivity on the dendritic tree. In particular, we would like to quantify not just which neurons are connected to a given cell, but also where these synaptic inputs are on the postsynaptic dendritic tree, and with what strength (Fig. 4.1). Such a technique would help in addressing a variety of open questions on the localization and maintenance of synaptic plasticity [Sjostrom et al., 2008], and would facilitate the study of nonlinear dendritic computations.

To achieve this goal, we can combine the ability to stimulate individual presynaptic neurons with high temporal resolution (either electrically or optically) and to simultaneously image postsynaptic neurons at subcellular spatial resolution. In particular, we can use two available, complementary types of data to obtain the best possible estimates:

1. Anatomical measurements of the postsynaptic neuron's shape and dendritic arborization. This provides a backbone on which we can build a dynamical model of the postsynaptic cell.

2. Voltage-sensitive fluorescence, observed at subcellular resolution. Modern imaging methods can access small dendritic structures and allow rapid sampling from many spatial locations [Reddy and Saggau, 2005; Iyer et al., 2006; Vucinic and Sejnowski, 2007; **?**]. This provides access to the key dynamical variable of interest, the spatiotemporal subthreshold voltage.

Since current voltage imaging technologies have relatively low signal-to-noise ratio (SNR) [Djurisic et al., 2004; Dombeck et al., 2004; Sacconi et al., 2006; Nuriya et al., 2006; Canepari

[1] This and the following section borrow heavily from the text of [Pakman et al., 2012].

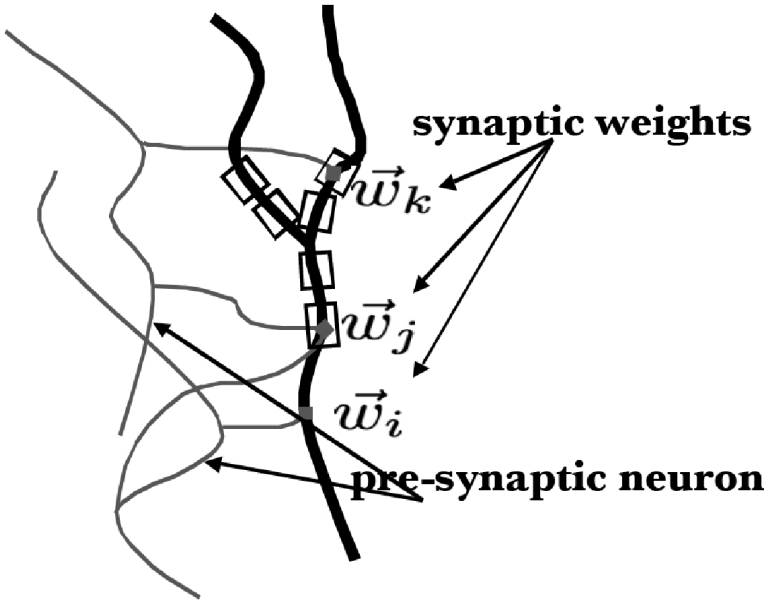


Figure 4.1: Schematic of method. By observing a noisy, subsampled spatiotemporal voltage signal on the dendritic tree, simultaneously with the presynaptic neuron's spike train, we can infer the strength of a given presynaptic cell's inputs at each location on the postsynaptic cell's dendritic tree.

et al., 2007; Milojkovic et al., 2007; Fisher et al., 2008; Djurisic et al., 2008; Canepari et al., 2008; **?**], we have to apply optimal filtering methods to exploit these measurements .

[Pakman et al., 2012] present fast methods to optimally filter such voltage measurements and infer synaptic weights. There the problem is formulated in a state-space model framework and builds on previously developed fast Bayesian methods [Huys et al., 2006; Huys and Paninski, 2009; Paninski and Ferreira, 2008; Paninski, 2010; Huggins and Paninski, 2012; **?**] for performing optimal inference of subthreshold voltage given noisy and incomplete observations. [Pakman et al., 2012] showed that these fast filtering methods can be combined with fast optimization methods from the sparse Bayesian literature [Efron et al., 2004b] to obtain a fast Bayesian solution to this synaptic estimation problem.

In inferring synaptic weights, an alternative to MAP with a regularizing prior is to take a fully Bayesian approach, for which there are several options. In such an approach, we choose from among various known sparsity-inducing priors, and more fully characterize

the resulting posterior distribution, ultimately yielding not only a point estimate for the synaptic weights, but also some measure of the uncertainty in our estimate: error bars, to put it simply.

In the next section we describe the dynamical model, following closely [Pakman et al., 2012]. Following that, we introduce the two families of priors we use to enforce sparsity in our synaptic weight estimates: the horseshoe model and spike-and-slab models. We then derive the key equations used to implement the inference procedure in each case. We conclude with the results of numerical experiments comparing the performance of these Bayesian methods against one another and against the method in [Pakman et al., 2012].

4.3 Dynamical model

We begin by describing the dynamical model adopted in [Pakman et al., 2012]. We assume that observations are available from a neuron with N compartments in which the passive cable dynamics and the observation equations are

$$V_{t+dt} = AV_t + WU_t + \epsilon_t, \qquad \epsilon_t \sim \mathcal{N}(0, \sigma^2 dt\mathbb{I}) \qquad t = 0, \ldots, T-1 \tag{4.1}$$

$$y_t = B_t V_t + \eta_t, \qquad \eta_t \sim \mathcal{N}(0, C_y\mathbb{I}) \qquad t = 1, \ldots, T\,. \tag{4.2}$$

In the first equation, V_t is an unobserved N-dimensional vector of compartment voltages at time t that evolves according to a discretized cable equation with timestep dt, perturbed by a Gaussian noise source ϵ_t. Assuming we can stimulate K presynaptic neurons in a controlled manner, U_t represents a K-dimensional vector of known presynaptic signals (the presynaptic spike times filtered by some fixed synaptic current filter). Finally, W is the $N \times K$ matrix of synaptic weights that we want to estimate.

We assume an experimental setting in which we simultaneously perform Z voltage observations at each discrete time t. (Z could vary with time, but to keep the notation manageable we will assume that Z is fixed here.) In the second equation, (4.2), y_t is an $Z-$dimensional vector of observations related instantaneously to V_t by the $Z \times N$ matrix B_t that specifies how the observations are performed. C_y is the covariance noise of the observations, which depends on the imaging apparatus used in each experiment. We assume

this covariance is proportional to the identity for simplicity, i.e. $C_y \propto I$, but this condition can also be easily relaxed.

This linear Gaussian model, with a passive (i.e. voltage independent) dynamic matrix A, can be a valid description for regimes with low network firing rate, so that the postsynaptic dendritic tree is in a subthreshold state. Furthermore, we assume that synaptic inputs are sufficiently small that the postsynaptic response may be treated using linear dynamics. (In an experimental setting we may enforce the subthreshold condition pharmacologically, by blocking voltage-gated sodium channels in the post-synaptic cell.)

On the other hand, real neural systems are known to depart from this linear, passive Gaussian regime. The noise can be non-Gaussian and strongly correlated (due, e.g., to unobserved spikes in the circuit), and the dynamics equation becomes non-linear when voltage dependent conductances and driving forces are taken into account. Also, for some measurement techniques, the observation equation may depart from the form (4.2). [Pakman et al., 2012] discuss some of these generalizations.

The log-likelihood $\log p(Y, V|W)$ can't be evaluated because it involves the unobserved voltages, but we can consider $p(Y|W)$. This quantity is Gaussian, since $p(Y, V|W)$ is Gaussian, so $\log p(Y|W)$ is a quadratic form $W^T MW + r^T W$, where M and r depend on Y. For details of the relationship between Y and M and r, see [Pakman et al., 2012]. In that work the authors take for W a lasso ('least absolute shrinkage and selection operator') prior

$$\log p(W|\lambda) = -\lambda \sum_{i,j} |W^{i,j}| + const. \tag{4.3}$$

where λ is a tuning parameter. This prior has the effect of sparsening the estimate $\hat{W}$. They show that maximum *a posteriori* inference in this setting is a concave problem, and they exploit certain features of the experimental setup to keep the computational cost of inference low. They modify this method (which they call LARS, for 'least angle regression', which is an algorithm for solving the lasso), introducing a prior with support only on the positive reals. (They call the method employing this prior LARS+.) This encodes into our model the phenomenon known as Dale's law, according to which all of a given neuron's synapses are either excitatory or inhibitory.

The numerical experiments reported in [Pakman et al., 2012] show that the LARS and

LARS+ inference methods for inferring the location and strength of synaptic connections are useful, at least when sampling in certain ways (i.e. when B_t has certain structure). They report that LARS+ performs better than LARS (and better than the ordinary least squares solution) and is able to learn the weights even under low SNR conditions.

4.4 Bayesian approaches

As alternatives to LARS and LARS+, we considered and implemented as priors over W the Bayesian Lasso of [Park and Casella, 2008], the Horseshoe prior introduced in [Carvalho et al., 2010], and spike-and-slab models like those described in [Mitchell and Beauchamp, 1988]. Ultimately, in the context of our problem, what we found that, among these alternatives, the most reliable estimates came from were spike-and-slab models. The Bayesian lasso did not perform well and we will not go into its details, but we will derive the key equations for the inference algorithms with the horseshoe and spike-and-slab models.

4.4.1 The horseshoe model

The horseshoe model for a very simple linear regression problem is defined and described in [Carvalho et al., 2010]. Here we first define the horseshoe model, following closely the description in [Carvalho et al., 2010], and then we write out the full conditional distributions necessary to run the Gibbs sampler. Last, we lay out the precise algorithm.

In the horseshoe model we observe a p-dimensional vector y such that $y|\theta \sim \mathcal{N}(\theta, \sigma^2\mathbb{I})$, and we suppose that

$$\theta_i|\lambda_i \sim \mathcal{N}(0, \lambda_i^2), \tag{4.4}$$

$$\lambda_i|\tau \sim \mathrm{C}^+(0, \tau), \tag{4.5}$$

$$\tau|\sigma \sim \mathrm{C}^+(0, \sigma) \tag{4.6}$$

where $\mathrm{C}^+(0, a)$ is the standard positive half-Cauchy distribution with scale parameter a. One measure of the amount of shrinkage of θ_i toward zero is $E(\kappa_i|y)$, where we have defined the parameter

$$\kappa_i \equiv \frac{1}{1 + \lambda_i^2}. \tag{4.7}$$

The model get its name from the horseshoe-shaped prior for κ_i that is implied by the half-Cauchy prior on λ_i. That is, the model is such that *a priori* the probability of very little shrinkage or very severe shrinkage is high, with the probability of intermediate shrinkage much lower, the aim being to differentiate between signal and noise and to attenuate the latter while leaving the signal alone.

It can be readily be shown that joint distribution over the variables in the model is

$$p(y,\theta,\lambda,\tau,\sigma) \propto \frac{1}{\sigma^{p+1}} \frac{1}{1+\left(\frac{\tau}{\sigma}\right)^2} \frac{1}{\tau^p} \prod_i^p \frac{1}{1+\lambda_i^2} \frac{1}{\lambda_i} \exp\left\{\frac{-\theta_i^2}{2\lambda_i^2\tau^2}\right\} \exp\left\{\frac{-(y_i-\theta_i)^2}{2\sigma^2}\right\} \tag{4.8}$$

$$\propto \frac{1}{\sigma^{p+1}} \frac{1}{1+\left(\frac{\tau}{\sigma}\right)^2} \frac{1}{\tau^p} \prod_i^p \frac{1}{1+\lambda_i^2} \frac{1}{\lambda_i} \exp\left\{\left(\theta_i - \frac{\lambda_i^2\tau^2 y_i}{\sigma^2+\lambda_i^2\tau^2}\right)^2 \Big/ \left(\frac{2\lambda_i^2\tau^2\sigma^2}{\sigma^2+\lambda_i^2\tau^2}\right)\right\} \tag{4.9}$$

The full conditional for θ is the obvious Gaussian distribution. The full conditional for λ_i is:

$$p(\lambda_i|y,\theta,\tau,\sigma) \propto \frac{1}{1+\lambda_i^2} \frac{1}{\lambda_i} \exp\left\{\frac{-\theta_i^2}{2\lambda_i^2\tau^2}\right\} d\lambda_i \tag{4.10}$$

As for λ_i, make the change in variables $u_i = \lambda_i^2$:

$$p(u_i|y,\theta,\tau,\sigma) \propto \frac{1}{1+u_i} \frac{1}{u_i} \exp\left\{\frac{-\theta_i^2}{2\tau^2 u_i}\right\} du_i \tag{4.11}$$

To sample from the full conditional distribution for u_i we employ rejection sampling with an envelope distribution. Such a sampling scheme is based on the observation that if $X \sim f(x)$ and $g(x)$ – the "envelope" – is a density function such that $f(x) \leq Mg(x)$ for a constant $M \geq 1$, then to sample $X \sim f$ we can generate $Y \sim g$ and $U \sim Uniform([0, Mg(y)])$ until $0 < u < f(y)$ [Robert and Casella, 2005]. For an envelope we use the Inverse Gamma distribution with $\alpha = 1$ and $\beta = \frac{\theta_i^2}{2\tau^2}$. We sample u_i from this distribution and transform back to $\lambda_i = \sqrt{u_i}$.

$$p(\tau|y,\theta,\lambda,\sigma) \propto \frac{1}{1+\left(\frac{\tau}{\sigma}\right)^2} \frac{1}{\tau^p} \exp\left\{\frac{1}{2}\sum_i \frac{-\theta_i^2}{\lambda_i^2} \frac{1}{\tau^2}\right\} d\tau \tag{4.12}$$

As for τ, make the change in variables $u = \left(\frac{\tau}{\sigma}\right)^2$:

$$p(u|y,\theta,\lambda,\sigma) \propto \frac{1}{1+u} \frac{1}{u^{(p+1)/2}} \exp\left\{\frac{1}{2\sigma^2}\sum_i \frac{-\theta_i^2}{\lambda_i^2} \frac{1}{u}\right\} du \tag{4.13}$$

Again we make use of rejection sampling with an envelope distribution. For an envelope we use the Inverse Gamma distribution with $\alpha = (p+1)/2$ and $\beta = \frac{1}{2\sigma^2}\sum_i \frac{\theta_i^2}{\lambda_i^2}$. We sample u from this distribution and transform back to $\tau = \sigma\sqrt{u}$.

The full conditional for σ is:

$$p(\sigma|y,\theta,\lambda,\tau) \propto \frac{1}{\sigma^{p+1}} \frac{1}{1+\left(\frac{\tau}{\sigma}\right)^2} \exp\left\{\frac{-1}{2\sigma^2}\sum_i^p (y_i-\theta_i)^2\right\} \tag{4.14}$$

A change of variables $u = \left(\frac{\tau}{\sigma}\right)^2$ brings this to a form that we might think to envelope with a Gamma density:

$$p(u|y,\theta,\lambda,\tau) \propto \frac{u^{(p-2)/2}}{1+u} \exp\left\{\frac{-1}{2\tau^2}\sum_i^p (y_i-\theta_i)^2 u\right\} \tag{4.15}$$

Sample u from this distribution and then transform back to $\sigma = \frac{\tau}{\sqrt{u}}$. The full conditional distribution over σ is derived in the same way and is closely related.

It turns out that for the full conditional distributions for λ and τ (and σ), one cannot feasibly draw samples using rejection sampling with the envelope distributions we have mentioned because the acceptance rates are very low. But we can rejection sample from these conditionals by alternating between two different envelopes. The two envelopes are inverse Gamma densities (Gamma in the case of σ whose α parameters differ by one. Basically, one envelope works well (has high acceptance rate) for the target density half of the time, and the other works well the other half of the time: they complement one another.

Algorithm 1 Horseshoe Gibbs sampler

Initialize $\sigma_g = 1$, $\tau_g = 1$, $\lambda_{g,i} \sim \mathrm{C}^+(0, \tau)_g$, and $\theta_{g,i} \sim \mathcal{N}(0, \lambda_{g,i}^2)$ for $i = 1, 2, \ldots, p$, p the dimensionality of the data y.

for N_{iter} iterations **do**

Sample $\theta_{g,j}$, $j = 1, 2, \ldots, p$:

Set $m_j = \lambda_{g,j}^2 \tau_g^2 y_j \Big/ \left(\sigma_g^2 + \lambda_{g,j}^2 \tau_g^2\right)$.

Set $s_j = \lambda_{g,j}^2 \tau_g^2 \sigma_g^2 \Big/ \left(\sigma_g^2 + \lambda_{g,j}^2 \tau_g^2\right)$.

Draw $\theta_g \sim \mathcal{N}(m, s^2)$.

Tally θ_g.

Sample $\lambda_{g,j}$, $j = 1, 2, \ldots, p$:

Set $a = (p-1)/2$

Set $b = \theta_{g,j}^2 \big/ \left(2\tau_g^2\right)$.

Sample $L \sim p(L = x) \propto x^{-a}(1+x)^{-1}e^{-bx}$ by rejection sampling with envelope distributions $Inverse\ Gamma(a, b)$ and $Inverse\ Gamma(a-1, b)$, switching envelope distribution after each rejection.

Set $\lambda_{g,j} = \sqrt{L}$.

Sample τ_g:

Set $a = 2$

Set $b = \sum_j \theta_{g,j}^2 / (2\sigma_g^2 \lambda_{g,j}^2)$.

Sample $T \sim p(T = x) = x^{-a}(1+x)^{-1}e^{-bx}$ by rejection sampling with envelope distributions $Inverse\ Gamma(a, b)$ and $Inverse\ Gamma(a-1, b)$, switching envelope distribution after each rejection.

Set $\tau_g = \sqrt{T}$.

Sample σ_g:

Set $a = (p-1)/2$

Set $b = \sum_j (y_j - \theta_j)^2 / (2\tau^2)$.

Sample $S \sim p(S = x) = x^{a}(1+x)^{-1}e^{-bx}$ by rejection sampling with envelope distributions $Gamma(a, b)$ and $Gamma(a-1, b)$, switching envelope distribution after each rejection.

Set $\sigma_g = \sqrt{S}$.

end for

Algorithm 1 is written for the model as it is defined in [Carvalho et al., 2010]. However, our problem does not involve data generated from a Gaussian draw from the vector to be estimated. It can be shown that with minor modification the above algorithm is still useful. In particular, to sample W_g (corresponding to θ_g in the algorithm) we draw

$$W_g \sim \mathcal{N}(A^{-1}r/\sigma^2, A^{-1}) \tag{4.16}$$

with

$$A \equiv \frac{1}{\sigma^2}M + \frac{1}{\tau^2}diag(\lambda \otimes \lambda). \tag{4.17}$$

Also, σ in the original model is introduced in the full data likelihood, but in the synaptic model of [Pakman et al., 2012] we actually restrict our focus to the W-dependent terms of the likelihood for computational reasons. So we ignore σ and instead choose τ by empirical Bayes, using expectation-maximization to compute the estimate.

4.4.2 Spike-and-slab models

A spike-and-slab model of a single variable w consists of a prior that is a mixture of a continuous distribution (a centered Gaussian, for instance) and a delta distribution centered at the origin. In the first spike-and-slab model that we will consider here, the density of w is

$$f(w) = (1-a)\cdot\delta(w) + a\cdot\mathcal{N}(0,\tau^2)[w] \tag{4.18}$$

where $0 \leq a \leq 1$ and τ^2 is the variance of the Gaussian distribution, whose density we abbreviate in the rightmost term. The finite support at zero is consistent with our intuition that synaptic weights often are exactly zero, namely where there is no synapse nearby at all. Inference with the other priors we considered would never yield weight estimates of exactly zero.

We implemented two spike-and-slab priors: one with the density (4.18) and the other with the same density except that we replaced the Gaussian mixture component with a truncated Gaussian distribution:

$$f(w) = \begin{cases} (1-a)\cdot\delta(w) + a\cdot 2\mathcal{N}(0,\sigma^2)[w] & \text{if } w \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.19}$$

This second prior, with support only on non-negative weight values, is consistent with Dale's law, according to which a given neuron's synapses are either all inhibitory or all exictatory: i.e., the synaptic weights all have the same sign [Gerstner and Kistler, 2002]. This assumes that all of the synapses onto the neuron in question connect it to a single other neuron.

Inference in both cases consists in Gibbs sampling the posterior of the weights given the data $D = \{M, r\}$ and under the spike-and-slab prior. Not surprisingly, the formulas involved in the unconstrained case are arrived at more readily than those of the constrained sampler. We outline here derivation of the spike-and-slab sampler algorithms. We then show a comparison of the performance of our Bayesian methods, alongside the performance of the LARS and LARS+ algorithms. We also treat inference with a log-normal slab, which is more in keeping with empirical distributions of synaptic weights.

4.4.2.1 The unconstrained spike-and-slab model

We can draw from the spike-and-slab prior itself, i.e. the density (4.18), by first drawing a Bernoulli sample with success rate a and then drawing from the Gaussian $\mathcal{N}(0, \sigma^2)$ if the result is success, and drawing from the delta distribution (i.e. picking zero) if the result is failure. Since the posterior is simply this mixture prior times the likelihood, samples from the posterior can be drawn in just the same way, but with a different success rate α_i for each weight and with a different, generally noncentered, Gaussian mixture component. In the following we find both the sparisity parameter and the Gaussian component of the posterior.

4.4.2.1.1 Finding the sparsity parameter Define $S \in \mathbb{R}^N$ to be a vector of such Bernoulli draws, such that

$$s_i|\alpha_i \sim Bernoulli(\alpha_i) \tag{4.20}$$

$$w_i|s_i, \tau^2 \sim \begin{cases} \mathcal{N}(0, \tau^2) & \text{if } s_i = 1 \\ \delta(w_i) & \text{if } s_i = 0 \end{cases} \tag{4.21}$$

By a straightforward application of Bayes rule it can be shown that

$$\alpha_i = p(s_i = 1|S_{-i}, D) = \frac{1}{1 + (1-a)/(a \cdot R)} \tag{4.22}$$

where

$$R = \frac{p(D|S_{-1}, s_i = 1)}{p(D|S_{-i}, s_i = 0)} \tag{4.23}$$

where we have defined S_{-i} to be S with the i^{th} element deleted. (The same subscript has another meaning for the vectors r and W and for the matrix M, which is defined below.)

In fact, we will not need α_i itself. We would use α_i by drawing $s_i|S_{-i}, D$ from its Bernoulli distribution, in the usual Gibbs sampler. However, following the ideas outlined in [Liu, 2008], we increase the efficiency of the Gibbs sampler by "Metropolizing" it: at sweep g of the sampler, we change the value of s_i with acceptance ratio

$$p\left(s_i^{(g+1)} = 1 - s_i^{(g)}\right) = \min\left\{1, \frac{p\left(s_i = 1 - s_i^{(g)}|S_{-i}, D\right)}{p\left(s_i = s_i^{(g)}|S_{-i}, D\right)}\right\} = \min\left\{1, \left(\frac{1-a}{a \cdot R}\right)^{2s_i^{(g)}-1}\right\} \tag{4.24}$$

Our aim, then, is to compute R. This can be achieved by writing both the numerator and denominator as integrals over W_{-i}, but there is a shorter route to the final expression that will suggest how to approach the same computation in the constrained model. This shorter route consists in viewing R as an expectation value. We can write

$$p(D|S_{-i}, s_i = 0) = \int dW p(D|W, S_{-i}, s_i = 0) p(W|S_{-i}, s_i = 0) \tag{4.25}$$

$$\propto \int dW_{-i} \exp\left\{-\frac{1}{2} W_{-i}^T A W_{-i} + r_{-i}^T W_{-i}\right\} \tag{4.26}$$

where $A = M_{-i} + \frac{1}{\tau^2}\mathbb{I}$ and M_{-i} is the matrix resulting from removing some of the rows and columns of M: those indexed by i and those with indices corresponding to elements of S that are zero. Similarly, W_{-i} and r_{-i} are the vectors W and r, respectively, with the i^{th} element removed, as well as those elements W_j and r_j such that $S_j = 0$.

Similarly,

$$p(D|S_{-i}, s_i = 1) \propto \int dw_i \int dW_{-i} \exp\left\{-\frac{1}{2} W_{-i}^T A W_{-i} + r_{-i}^T W_{-i}\right\} \cdot \tag{4.27}$$

$$\cdot \frac{1}{\sqrt{2\pi\tau^2}} \exp\left\{-\sum_{\substack{j:s_j=1 \\ j\neq i}} w_i w_j M_{ji} - \frac{1}{2} w_i^2 M_{ii} + r_i w_i - \frac{1}{2\tau^2} w_i^2\right\} \tag{4.28}$$

The proportionality constants in Equations (4.25) and (4.27) are the same. Then we have that

$$R = \frac{1}{\sqrt{2\pi\tau^2}} \int dw_i \mathbb{E}\left[e^{-w_i W_{-i}^T m_i}\right] \exp\left\{-\frac{1}{2} w_i^2 \left(M_{ii} + \frac{1}{\tau^2}\right) + r_i w_i\right\} \tag{4.29}$$

where m_i is the i^{th} column of M with removed all the elements corresponding to zero elements of S as well as the i^{th} element. Evaluating a couple of straightforward Gaussian integrals yields that R, in full, is given by

$$R = \sqrt{\frac{1}{\tau^2 \left|M_{ii} + \frac{1}{\tau^2} - m_i^T A^{-1} m_i\right|}} \exp\left\{\frac{\left(m_i^T A^{-1} r_{-i} - r_i\right)^2}{M_{ii} + \frac{1}{\tau^2} - m_i^T A^{-1} m_i}\right\} \tag{4.30}$$

and we can find α_i via Equation (4.22).

4.4.2.1.2 Finding the Gaussian mixture component For selecting the variance τ^2 of the Gaussian slab component of the prior, we use an expectation-maximization algorithm. We present both E and M steps together:

$$\tau_{t+1}^2 = \arg\max_{\tau^2} E(\log p(D, W, S|\tau^2))_{\tau_t^2} \tag{4.31}$$

$$= \arg\max_{\tau^2} E(\log p(W|S, \tau^2))_{\tau_t^2} \tag{4.32}$$

$$= \arg\max_{\tau^2} \sum_S \int da\, dW_S \left(-\frac{1}{2}|S| \log \tau^2 - \frac{W_S^T W_S}{2\tau^2}\right) p(W_S|D, S, \tau_t^2) p(S, a|D, \tau_t^2) \tag{4.33}$$

$$= \arg\max_{\tau^2} \frac{1}{J} \sum_{j=1}^{J} \int dW_{S_j} \left(-\frac{1}{2}|S_j| \log \tau^2 - \frac{W_{S_j}^T W_{S_j}}{2\tau^2}\right) p(W_{S_j}|D, S_j, \tau_t^2) \tag{4.34}$$

where the S_j are J samples of $\{S, a\}$ from the posterior $p(S, a|D, \tau_t^2)$ and can be obtained using the collapsed Gibbs sampler. For each S_j, we take Q samples $\{W_{S_j,i}\}$, $i = 1, \cdots, Q$ from

$$p(W_{S_j}|D, S_j, \tau_t^2) \propto \exp\left\{-\frac{1}{2} W_{S_j}^T \left(M + \frac{1}{\tau^2}\right) W_{S_j} + r^T W_{S_j}\right\} \qquad W \geq 0 \tag{4.35}$$

and we get

$$\tau_{t+1}^2 \approx \arg\max_{\tau^2} \frac{1}{JQ} \sum_{j=1}^{J} \sum_{i=1}^{Q} \left\{-\frac{1}{2}|S_j| \log \tau^2 - \frac{W_{S_j,i}^T W_{S_j,i}}{2\tau^2}\right\} \tag{4.36}$$

$$= \frac{1}{KQ} \sum_{j=1}^{J} \sum_{i=1}^{Q} W_{S_j,i}^T W_{S_j,i} = \left\langle W^T W \right\rangle / \left\langle |S| \right\rangle \tag{4.37}$$

where the averages are over Gibbs sweeps and where we defined

$$K = \sum_{j=1}^{J} |S_j| \tag{4.38}$$

We take this same approach to update the sparsity parameter a. The update is

$$a = \langle |S| \rangle / N \tag{4.39}$$

4.4.2.1.3 Sampling sparsity and weights vectors In summary, then, given a value of a, we sample S from the posterior by, for each component s_j, computing R as in equation (4.30), in turn computing α_j, and sampling $s_j \sim Bernoulli(\alpha_j)$. The details are shown in Algorithm (2). Note that whereas the prior $p(S|a)$ factorizes into independent Bernoulli components of S, the posterior $p(S|W, D, a)$ does not factorize. Different components of S depend on one another. This manifests in Equation (4.30) in that the indices of the components of, for instance, r that make up r_{-i} are the indices of the components of S_{-i} that are zero.

Algorithm 2 Sampling the sparsity vector S in the unconstrained spike-and-slab model

Given the current sample S,

for each component j of S **do**

Compute m_j, the j^{th} column of M.

Compute $\mathcal{I}_S$, the set of indices i for which $s_i = 0$.

Compute M_{-j}, the matrix M with removed rows and columns with index in $\mathcal{I}_S$.

Compute r_{-j} by removing from r each component with index in $\mathcal{I}_S$ and also the j^{th} component.

Compute $A = M_{-j} + \mathbb{I}/\tau^2$.

Compute $R = \left(\tau^2 \left| M_{jj} + 1/\tau^2 - m_j^T A^{-1} m_j \right|\right)^{-1/2} \cdot$
$\cdot \exp\left\{ \left(m_j^T A^{-1} r_{-i} - r_i\right)^2 \Big/ \left(M_{jj} + 1/\tau^2 - m_j^T A^{-1} m_j\right) \right\}$

Compute $\alpha_j = \left(1 + \frac{1-a}{a} R\right)^{-1}$.

Draw $s_j \sim Bernoulli(\alpha_j)$.

end for

Once we have S we need to sample $W|S, D$. By Bayes rule,

$$p(W|S, D) \propto p(D|W, S)p(W|S) \tag{4.40}$$

which is just the integrand in equation (4.26). Call W_1 the vector W with all zeros deleted, and similarly define r_1 and M_1 and A_1. Completing the square, we see that

$$W_1|S, D \sim \mathcal{N}(A_1^{-1}r_1, A_1^{-1}) \tag{4.41}$$

Since we have analytical expressions for the sufficient statistics of $p(W|S, D)$, we can Rao-Blackwellize our Gibbs sampler, recording at each step the mean and covariance of $W|S, D$, which should yield an estimator with smaller variance [Robert and Casella, 2005]. This computation is detailed in Algorithm (3). Once again, whereas $p(W|S)$ factorizes into independent components of W, $p(W|S, D)$ is a correlated normal density.

Algorithm 3 Rao-Blackwell step in unconstrained model: record sufficient statistics of $W|S, D$

Given the current sample S,

Compute M_1, the matrix M with entries of rows and columns with index in $\mathcal{I}_S$ set to zero.

Compute $A_1 = M_1 + \mathbb{I}/\tau^2$.

Record $\mu = A_1^{-1}r_1$ and $C = A_1^{-1}$.

The whole inference algorithm for the unconstrained model is detailed in Algorithm (4).

4.4.2.2 The constrained spike-and-slab model

The constrained prior of Equation (4.19) encodes our assumption that all synaptic weights should have the same sign, i.e. Dale's law. This prior introduces the challenge of sampling from high-dimensional truncated distributions (truncated normals, in our case). Inference with the constrained prior follows the same outline as in the unconstrained case, but the sampling of S and W is handled in a different way. The empirical Bayes method of setting hyperparameters a and τ^2 is unchanged.

Algorithm 4 Estimation in the unconstrained spike-and-slab model

Choose starting values for variance τ^2 and sparsity parameter a.

for $N_{E.B.}$ iterations (or until convergence of τ^2 and a) **do**

Initialize S, say, by drawing each element from $Bernoulli(a)$.

for N_G Gibbs sweeps after N_B burn-in cycles **do**

Sample S according to Algorithm (2).

Sample W according to Algorithm (3). Record W only after burn-in cycles.

end for

Set $\tau^2 = \left\langle W^T W \right\rangle / \left\langle |S| \right\rangle$ and $a = \left\langle |S| \right\rangle / N$, where the averages are over Gibbs sweeps.

end for

The most obvious difference in sampling S and W in the constrained setting is that, instead of sampling all the components of S and then drawing a sample $W|S$ from a multivariate normal distribution, here we sample block-wise component pairs (s_i, w_i), as suggested in [Mohamed et al., 2011]. As opposed to the unconstrained case, here we cannot easily sample from the marginal $p(s_i|S_{-i}, D)$ because integrating over W to compute R is intractable.

4.4.2.2.1 Blockwise sampling of sparsity and weight components The main idea here is that in each Gibbs sweep we cycle through pairs (s_i, w_i) of corresponding components of S and W, first sampling $s_i|S_{-i}, W_{-i}, D$, and then sampling $w_i|s_i, S_{-i}, W_{-i}, D$. We write

$$p(s_i, w_i|S_{-i}, W_{-i}, D) = p(s_i|S_{-i}, W_{-i}, D) \cdot p(w_i|s_i, S_{-i}, W_{-i}, D) \tag{4.42}$$

The first factor is Bernoulli and we can compute the success rate α_i similarly to how we did in the unconstrained setting. The second factor is a delta function when $s_i = 0$, and is a truncated normal when $s_i = 1$, as we show next. We can sample efficiently from one-dimensional truncated normal distributions, and so we can effectively Rao-Blackwellize this estimator too; for each sample $s_i|S_{-i}, W_{-i}, D$, we draw multiple samples of $w_i|s_i, S_{-i}, W_{-i}, D$.

4.4.2.2.2 Finding the sparsity parameter To compute $\alpha_i = p(s_i = 1|S_{-i}, W_{-i}, D)$ in the constrained setting, once again we start by considering the ratio

$$\frac{p(s_i = 1|S_{-i}, W_{-i}, D)}{p(s_i = 0|S_{-i}, W_{-i}, D)} = \frac{p(D|S_{-i}, s_i = 1, W_{-i})}{p(D|S_{-i}, s_i = 0, W_{-i})} \cdot \frac{p(s_i = 1|S_{-i}, W_{-i})}{p(s_i = 0|S_{-i}, W_{-i})} \tag{4.43}$$

$$= \frac{p(D|S_{-i}, s_i = 1, W_{-i})}{p(D|S_{-i}, s_i = 0, W_{-i})} \cdot \frac{a}{1-a} \tag{4.44}$$

$$\equiv \rho \cdot \frac{a}{1-a} \tag{4.45}$$

Since we are conditioning on W_{-i} now, computing ρ is simpler than computing R of the unconstrained setting. Consider first the denominator of ρ:

$$p(D|S_{-i}, s_i = 0, W_{-i}) = \int dw_i p(D|S_{-i}, s_i = 0, W_{-i}, w_i) \cdot p(w_i|S_{-i}, s_i = 0, W_{-i}) \tag{4.46}$$

$$= p(D|S_{-i}, s_i = 0, W_{-i}, w_i = 0) \tag{4.47}$$

$$= p(D|W_{-i}, w_i = 0) \tag{4.48}$$

$$\propto \exp\left\{-\frac{1}{2}W_{-i}^T M_{-i} W_{-i} + r_{-i}^T W_{-i}\right\} \tag{4.49}$$

The numerator of ρ is

$$p(D|S_{-i}, s_i = 1, W_{-i}) = \int dw_i p(D|S_{-i}, s_i = 1, W_{-i}, w_i) \cdot p(w_i|S_{-i}, s_i = 1, W_{-i}) \tag{4.50}$$

$$= \int dw_i p(D|S_{-i}, s_i = 1, W_{-i}, w_i) \mathcal{N}'(0, \tau^2)[w_i] \tag{4.51}$$

$$\propto \int dw_i \exp\left\{-\frac{1}{2}W_{-i}^T M_{-i} W_{-i} + r_{-i}^T W_{-i}\right\} \cdot \tag{4.52}$$

$$\cdot \frac{2}{\sqrt{2\pi\tau^2}} \exp\left\{-w_i W_{-i}^T m_i - \frac{1}{2}w_i^2 M_{ii} + r_i w_i - \frac{1}{2\tau^2}w_i^2\right\} \cdot \tag{4.53}$$

$$\cdot \mathbb{I}(w_i > 0) \tag{4.54}$$

where $\mathcal{N}'(\mu, \sigma^2)$ is a normal distribution truncated to the nonnegative reals with mean μ and variance σ^2. The first factor of the numerator cancels with the denominator. Completing the square on what remains and performing the integral, we arrive at

$$\rho = \frac{2}{\sqrt{A_{ii}\tau^2}} e^{-k/2} \Phi\left(h\sqrt{A_{ii}}\right) \tag{4.55}$$

where

$$h = \left(W_{-i}^T m_i - r_i\right)/A_{ii} \tag{4.56}$$

$$k = -\left(W_{-i}^T m_i - r_i\right)^2/A_{ii} \tag{4.57}$$

$$A_{ii} = M_{ii} + 1/\tau^2 \tag{4.58}$$

and $\Phi(x)$ is the cumulative distribution function of a standard normal distribution, evaluated at x.

Once we have ρ, we compute α_i and sample s_i. As for w_i, we want to sample from

$$p(w_i|s_i = 1, S_{-i}, W_{-i}, D) \propto p(D|s_i = 1, S_{-i}, w_i, W_{-i}) \cdot p(w_i|s_i = 1, S_{-i}, W_{-i}) \tag{4.59}$$

We recognize the first factor on the right as the integrand of the numerator of ρ. The second factor on the right is simply the truncated normal $p(w_i|s_i = 1, S_{-i}, W_{-i}) = p(w_i|s_i = 1) = \mathcal{N}'(0, \tau^2)$, leaving us with

$$w_i \sim \mathcal{N}'\left(h, A_{ii}\right) \tag{4.60}$$

with h and A_{ii} defined as above. The inference algorithm in the constrained setting is detailed in Algorithms (5) and (6).

4.4.2.3 Spike-and-slab with log-normal slab

Instead of a Gaussian (or truncated Gaussian) slab, consider inference of synaptic weights with a log-normal prior, which is more consistent with experimental evidence:

$$s_i|\alpha_i \sim Bernoulli(\alpha_i) \tag{4.61}$$

$$w_i|s_i, \mu, \tau^2 \sim \begin{cases} \frac{1}{x\sqrt{2\pi\tau^2}} \exp\left\{-\frac{(\log x - \mu)^2}{2\tau^2}\right\} & \text{if } s_i = 1 \\ \delta(w_i) & \text{if } s_i = 0 \end{cases} \tag{4.62}$$

As with the constrained Gaussian slab, we Gibbs sample pairs (s_i, w_i). We want to sample from

$$p(s_i, w_i|S_{-i}, W_{-i}, D) = p(s_i|S_{-i}, W_{-i}, D) \cdot p(w_i|s_i, S_{-i}, W_{-i}, D) \tag{4.63}$$

Algorithm 5 Sampling a pair (s_j, w_j) of sparsity and weight components

Given the current sample S_{-j} and W_{-j},

Compute m_j, the j^{th} column of M.

Compute $\mathcal{I}_S$, the set of indices i for which $s_i = 0$.

Compute M_{-j}, the matrix M with removed rows and columns with index in $\mathcal{I}_S$.

Compute r_{-j} by removing from r each component with index in $\mathcal{I}_S$ and also the j^{th} component.

Compute $A = M_{-j} + \mathbb{I}/\tau^2$ and $A_{jj} = M_{jj} + 1/\tau^2$

Compute $h = (W_{-j}^T m_j - r_j)/A_{jj}$

Compute $k = -(W_{-j}^T m_j - r_j)^2/A_{jj}$

Compute $\rho = \frac{1}{\sqrt{2\pi\tau^2}} e^{-k/2} \left(1 - \Phi(-h\sqrt{A_{jj}})\right)$

Compute $\alpha_j = a/(a + (1-a)\rho)$.

Draw $s_j \sim Bernoulli(\alpha_j)$.

Draw $w_j \sim \mathcal{N}'(h, \sqrt{A_{jj}})$; sample many $w_j^{(l)}$ for effective Rao-Blackwellization.

Algorithm 6 Estimation in the constrained spike-and-slab model

Choose starting values for variance τ^2 and sparsity parameter a.

for $N_{E.B.}$ iterations (or until convergence of τ^2 and a) **do**

  Initialize S and W, say, by drawing from $p(S, W|a, \tau^2)$.

  for N_G Gibbs sweeps after N_B burn-in cycles **do**

    for each component i of S **do**

      Sample s_i and w_i according to Algorithm (5). Record w_i only after burn-in cycles.

    end for

  end for

  Set $\tau^2 = \left\langle W^T W \right\rangle / \left\langle |S| \right\rangle$ and $a = \left\langle |S| \right\rangle / N$, where the averages are over Gibbs sweeps.

end for

We cannot sample from $p_1(s_i) = p(s_i|S_{-i}, W_{-i}, D)$ exactly, but we can draw approximate samples – samples from $\tilde{p}_1(s_i)$, say – by making a Laplace approximation to the likelihood, as shown below. It is also not possible to draw exact samples from $p_2(w_i|s_i) = p(w_i|s_i, S_{-i}, W_{-i}, D)$. We will sample instead from a similar distribution $\tilde{p}_2(w_i|s_i)$ and accept these samples with a Metropolis-Hastings acceptance probability.

The distribution for which we will make a Laplace approximation is

$$p_2(w_i|s_i = 1) = p(w_i|s_i = 1, S_{-i}, W_{-i}, D) \tag{4.64}$$

$$= p(D|s_i = 1, S_{-i}, w_i, W_{-i}) \cdot p(w_i|s_i = 1) \tag{4.65}$$

$$\propto \frac{1}{w_i} \exp\left\{(r_i - W_{-i}^T m_i)w_i - \frac{1}{2}M_{ii}w_i^2 - \frac{(\log w_i - \mu)^2}{2\tau^2}\right\} \tag{4.66}$$

Differentiating, we arrive at

$$\left.\frac{dp_2(w_i|s_i = 1)}{dw_i}\right|_{w_i=w_0} = 0 \quad \Longrightarrow \quad \left(r_i - W_{-i}^T m_i\right) - M_{ii}w_0 - \frac{1}{\tau^2}(\log w_0 - \mu + \tau^2)\frac{1}{w_0} = 0 \tag{4.67}$$

The presence of the logarithm precludes a closed form solution for w_0. We can approximate $\log(1 + x)$ by the first term in its Taylor series (x) when x is small. We know that the mode of the log-normal distribution is $e^{\mu-\tau^2}$. We can imagine factoring out the mode from w_0, resulting in the logarithm of a number close to one. Unfortunately the deviation of w_0 from the mode is often substantial and the approximation doesn't hold. We have resorted to solving for w_0 numerically by Newton's method.

With w_0 in hand, we find the curvature of the peak of the true distribution:

$$A = -\frac{d^2}{dw_i^2}\log p_2(w_i|s_i = 1)\Big|_{w_i=w_0} \tag{4.68}$$

$$= -\frac{d^2}{dw_i^2}\left\{-\log Z - \log w_i + \left(r_i - W_{-i}^T m_i\right)w_i - \frac{1}{2}M_{ii}w_i^2 - \frac{(\log w_i - \mu)^2}{2\tau^2}\right\}\Big|_{w_i=w_0} \tag{4.69}$$

$$= -\frac{d}{dw_i}\left\{-\frac{1}{w_i} + \left(r_i - W_{-i}^T m_i\right) - M_{ii}w_i - \frac{1}{\tau^2}(\log w_i - \mu)\frac{1}{w_i}\right\}\Big|_{w_i=w_0} \tag{4.70}$$

$$= M_{ii} + \frac{1}{\tau^2 w_0^2}\left(1 - \log w_0 + \mu - \tau^2\right) \tag{4.71}$$

where Z is a normalizing constant.

Bringing everything together, we come to the Gaussian approximation of Equation (4.66):

$$p_2(w_i|s_i=1) \approx \tilde{p}_2(w_i|s_i=1) \equiv \sqrt{\frac{A}{2\pi}} \exp\left\{-\frac{A}{2}(w_i - w_0)^2\right\} \tag{4.72}$$

Then our approximate ρ is

$$\rho = \frac{p(D|S_{-i}, s_i=1, W_{-i})}{p(D|S_{-i}, s_i=0, W_{-i})} \tag{4.73}$$

$$= \frac{\int_0^\infty dw_i p(D|S_{-i}, s_i=1, W_{-i}, w_i) p(w_i|s_i=1)}{p(D|S_{-i}, s_i=0, W_{-i}, w_i=0)} \tag{4.74}$$

$$= \frac{1}{\sqrt{2\pi\tau^2}} \int_0^\infty dw_i \frac{1}{w_i} \exp\left\{\left(r_i - W_{-i}^T m_i\right) w_i - \frac{1}{2} M_{ii} w_i^2 - \frac{(\log w_i - \mu)^2}{2\tau^2}\right\} \tag{4.75}$$

$$\approx \frac{1}{\sqrt{2\pi\tau^2}} \int_0^\infty dw_i f(w_0) \exp\left\{-\frac{A}{2}(w_i - w_0)^2\right\} \tag{4.76}$$

where we have defined

$$f(w) \equiv \frac{1}{w} \exp\left\{\left(r_i - W_{-i}^T m_i\right) w - \frac{1}{2} M_{ii} w^2 - \frac{(\log w - \mu)^2}{2\tau^2}\right\} \tag{4.77}$$

Then

$$\rho = \frac{f(w_0)}{\sqrt{A\tau^2}} \Phi\left(\sqrt{A} w_0\right) \tag{4.78}$$

While approximate inference in the spike-and-slab model with log-normal slab is tractable, we do not present the results of experiments with this slab, as the results do not significantly differ from those of experiments with the truncated Gaussian slab. Moreover, as the results of the following section show, spike-and-slab estimates with the truncated Gaussian slab successfully recover synaptic weights, even when they are drawn from a log-normal distribution.

4.5 Results

Figures (4.2), (4.3), (4.4), and (4.5) display the performance of both of the spike-and-slab inference algorithms (S+S for the unconstrained case and S+S+ for the constrained case),

alongside the LARS and LARS+ estimates. The S+S+ estimates are the only to ever be exactly zero. All the estimates deteriorate as the level of noise increases. While the Bayesian methods offer natural error bars, none of them uniformly outperforms the LARS and LARS+ methods.

4.6 Conclusions

Inferring the location of synapses in dendritic trees is a central problem in neuroscience and is one step toward mapping the synaptic connectivity of the entire tree. Methods using a linear model of subthreshold voltage dynamics throughout a compartmentalized neuron and making use of the lasso have been successful in recovering synaptic weights from noisy observations of the system. While this represents a substantial step toward addressing the problem of locating synapses, the methods do not provide much in the way of any measure of the uncertainty of the estimates. By contrast, a major strength of Bayesian methods is that their estimates come with natural error bars. Instead of merely reporting the Gibbs sample mean of the posterior of our estimand given the data, we may more fully characterize the posterior distribution – at a minimum, reporting its variance – to give a more comprehensive statement of what the value of the estimand might be.

Here we explored several Bayesian methods: the Bayesian lasso (results not shown), the horseshoe of [Carvalho et al., 2010], and spike-and-slab models with and without positivity constraint. We showed that these Bayesian methods do not outperform LARS and LARS+, and LARS+ seems to be the best method of them all. Why do the Bayesian methods fall short? Ostensibly, LARS and LARS+ do a better job of pulling small signals to zero and leaving large signals alone. Whether another choice of prior might produce a Bayesian method with superior performance remains an open question.

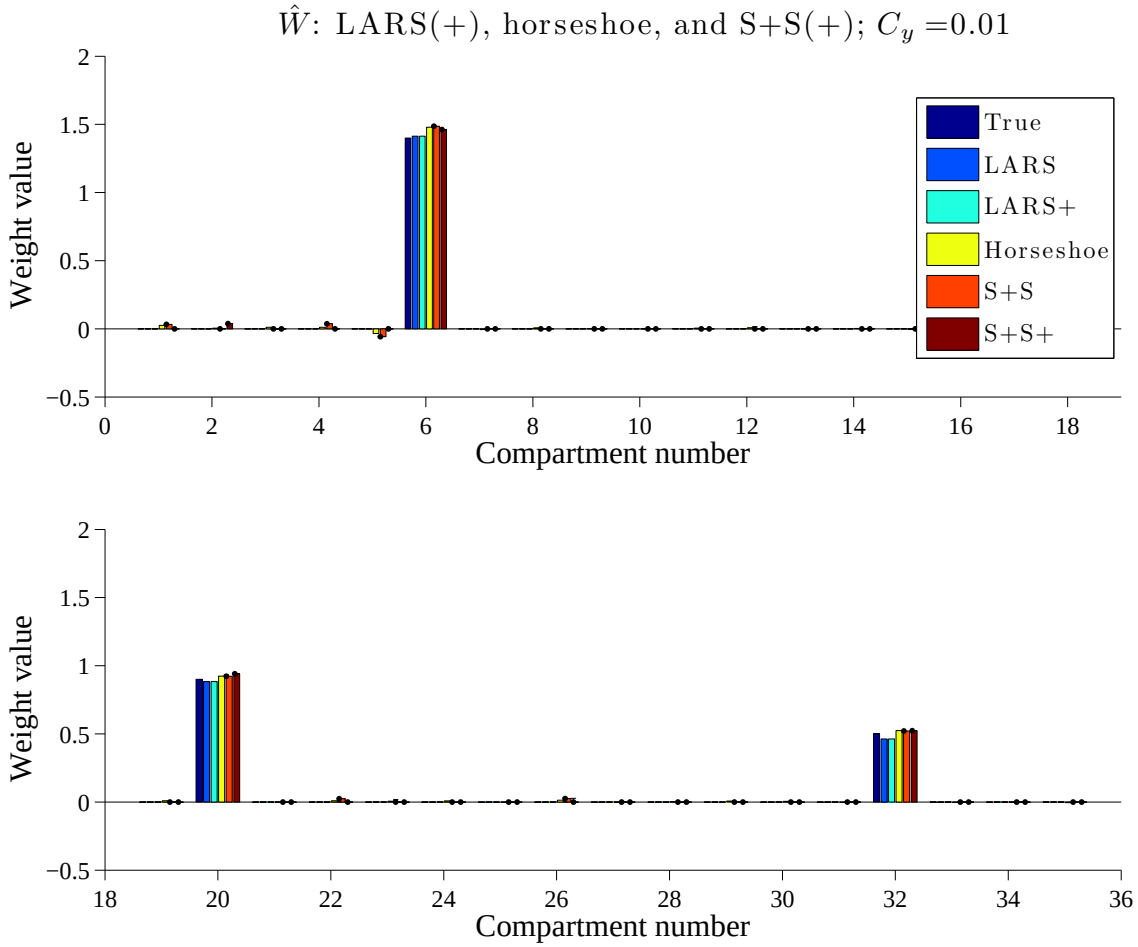


Figure 4.2: Estimates of synaptic weights for a small (35-compartment) neuron with true nonzero weights (blue) drawn from a log-normal distribution with shape parameter $\sigma^2 = 0.0625$ and log-scale parameter $\mu = 0$. Error bars on the spike-and-slab estimates at a given component show the quartiles of the set of Gibbs samples of that component (including samples from the spike). The observation noise in this case is $C_y = 0.01$. Spike-and-slab and horseshoe samplers ran for 1000 Gibbs sweeps after 100 burn-in sweeps.

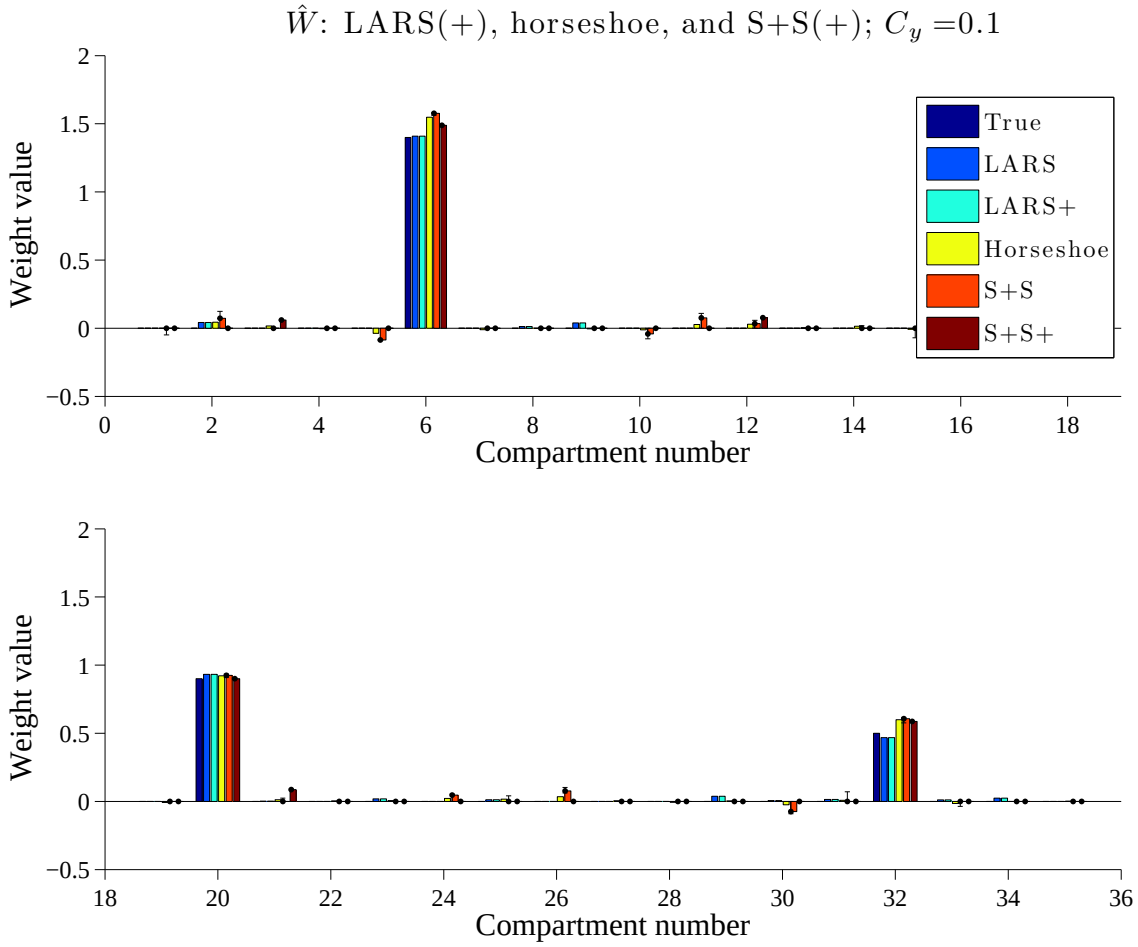


Figure 4.3: Estimates of synaptic weights for a small (35-compartment) neuron as in Figure (4.2) but with $C_y = 0.1$.

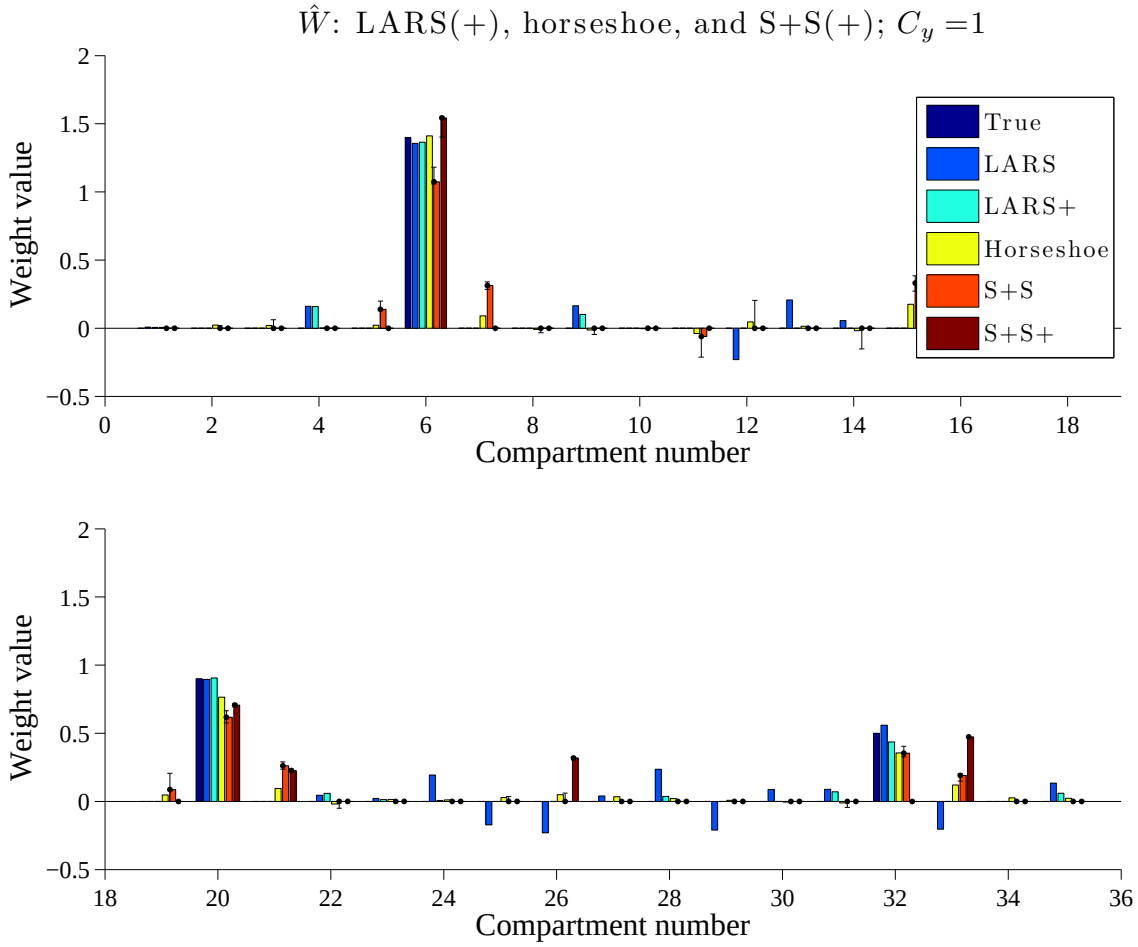


Figure 4.4: Estimates of synaptic weights for a small (35-compartment) neuron as in Figure (4.2) but with $C_y = 1$.

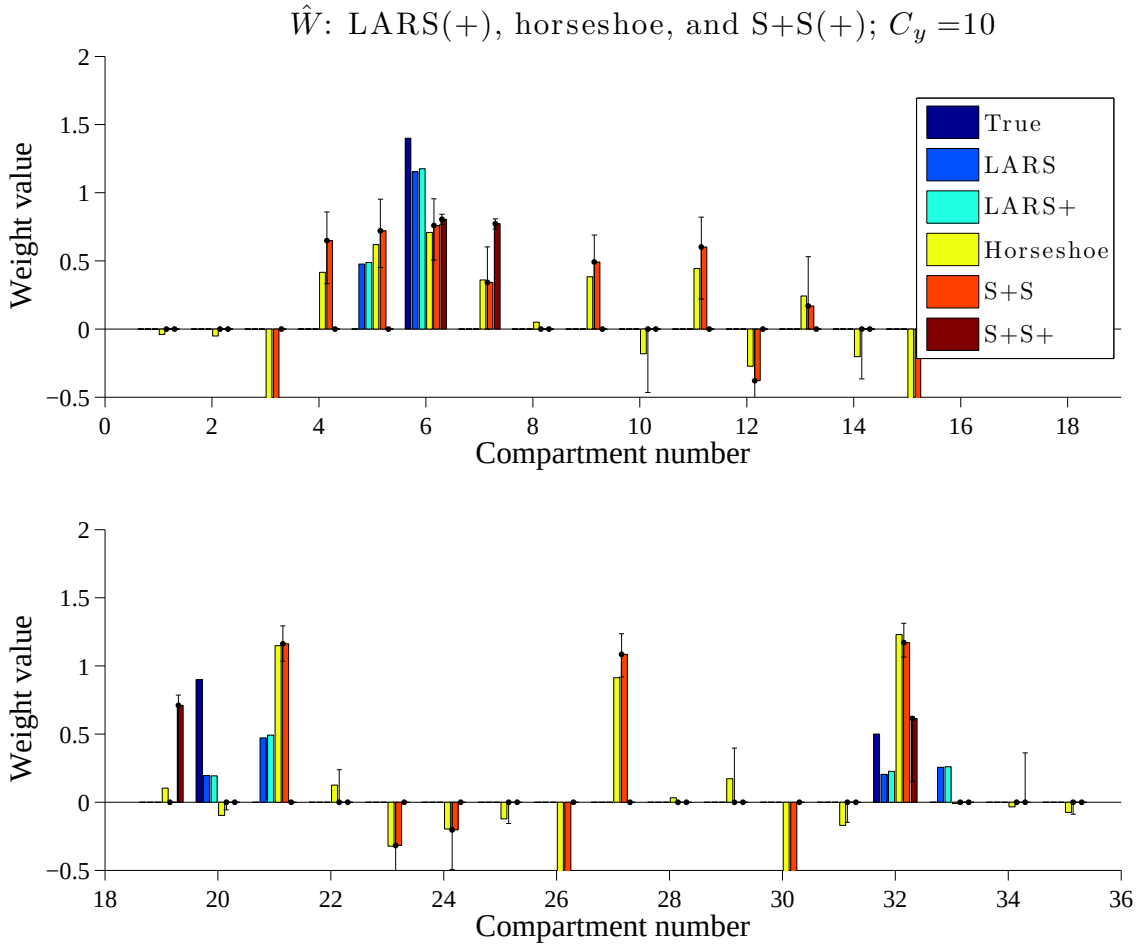


Figure 4.5: Estimates of synaptic weights for a small (35-compartment) neuron as in Figure (4.2) but with $C_y = 10$. In this case the horseshoe sampler was left to run for 10,000 Gibbs sweeps after 1,000 burn-in sweeps in case the poor performance were due partly to slow mixing.

Chapter 5

Inferring orientation selectivity maps with low-rank von Mises and related priors

5.1 Abstract

In many areas of the nervous system, different nearby neurons that respond to the same input (or, more generally, whose firing is related to the same covariate) do not respond in the same way as one another. In general, different neurons may be tuned to respond selectively to different particular features of the same input. Sometimes the distribution of this selectivity over neurons is, seemingly, spatially random. In other cases, though, selectivity varies smoothly with the location of the neuron in space. In this work we focus on the case of smoothly varying selectivity of angular variables. One such variable is orientation of a visual stimulus. In visual cortex, different neurons respond preferentially to different orientations of a presented stimulus. Another such variable is the phase of a system during a rhythmic motion. In particular, certain spinal neurons whose firing is correlated with the act of walking are such that any one neuron will fire most around a particular point in the the cyclic motion. In both the case of orientation selectivity and phase selectivity, selectivity has been found to vary rather smoothly with respect to the location of the neuron. In this

work, we explore two models – one of them one of the low-rank models of chapter 2 – for such phase maps and apply them to real and synthetic data. We find that here low-rank models are quite slow, which is to be expected given the loopy structure of the graphs that represent phase maps. A comparatively simple model is substantially faster and is successful at partially reconstructing phase maps from noisy or subsampled observations.

5.2 Introduction

We develop two models of smoothly varying phase selectivity maps. One is a loopy variant of the low-rank models described in Chapter 2 that exploits the self-conjugacy of the von Mises distribution – a distribution over angular variables. The other is a simpler model whose tractability also relies upon the self-conjugacy of the von Mises. In the following, we first recapitulate the essential ideas of low-rank models, specifically in the context of inference on directional random variables, i.e. variables that can be represented as unit vectors. Because Chapter 2 develops low-rank models only on tree (i.e. non-loopy) graphs, we provide a detailed derivation of the necessary forward backward variables and equations (and normalization constants) in this loopy case, for the block-wise approach that we take to necessarily approximate inference on this model. In a numerical experiment, we demonstrate the smoothness and patchiness of draws from the prior in this model. Then we move on to the simpler model. We define the model likelihood and prior and apply it to estimation of a spinal neuron phase selectivity map from subsampled data, showing that the estimate approaches the full observation, with respect to a natural measure of error.

5.3 Low-rank models revisited

Low-rank models are joint distributions whose structure allows for fast exact inference in settings where standard models such as hidden Markov models and linear Gaussian models are not applicable. We can perform fast inference on any low-rank model with a tree structure, i.e. without loops. Beyond that, low-rank inference may be used in the context of block-wise sampling of loopy distributions, which may speed up mixing of the sampler compared to a point-wise sampler.

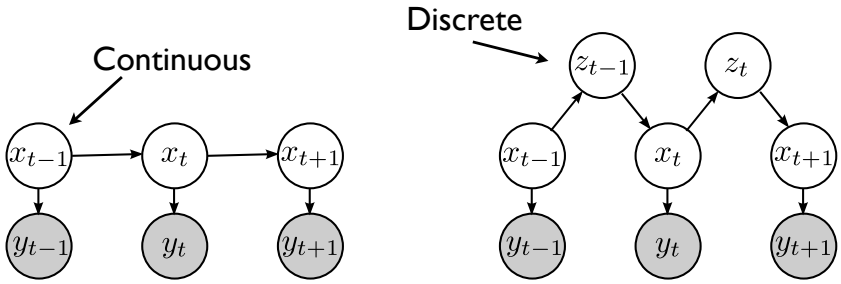


Figure 5.1: On the left, a general chain structured distribution of latent variables X and observations Y. On the right, such a model expressed in terms of discrete auxiliary variables Z.

Suppose, for simplicity, that we are interested in a distribution $p(X)$ over some latent variables X, whose dependence structure forms a chain. (More generally, consider the posterior of X given some observations Y.) Suppose that X is continuous but with non-Gaussian marginals, so that neither HMMs nor linear Gaussian models are appropriate. But suppose that the dependence structure of X can be expressed in terms of latent variables Z with some small discrete state space, where each neighboring pair of continuous x's is coupled through some discrete variable z, as shown in Figure 5.1. The graph on the left has the form

$$p(X|Y) = p(x_1) \prod_{t=1}^{T-1} \Big[p(x_{t+1}|x_t) \Big] \, p(y_t|x_t).$$

The graph on the right, as the distribution $p(X|Y)$, has the same form, but it can also be expressed as

$$p(X|Y) = p(x_1) \prod_{t=1}^{T-1} \left[\sum_{z_t}^{R_t} p(z_t|x_t) p(x_{t+1}|z_t) \right] p(y_t|x_t)$$

where we have replaced the transition kernel between x_t and x_{t+1} with a sum over z_t, which is a discrete variable with few states. This resembles the decomposition of a matrix into rank-one matrices, by singular value decomposition, for example, except that $p(z_t|x_t)$ and $p(x_{t+1}|z_t)$ are continuous functions, not vectors. Nonetheless, we can consider such a

potential function to be in some sense a truncated approximation to some more complicated or "high-rank" function, where R_t here is the rank of the truncated function, which is just the size of the state space of z_t. If the x's were discrete random variables, then this equation would represent a discrete Markov chain in which the transition densities are of rank R_t, and inference in this kind of Markov chain is relatively easy, since multiplication by low-rank matrices is relatively cheap.

The key idea here is that exact inference of the Markov chain X is tractable even in the general, non-discrete case. In these distributions, X can be marginalized out, leaving a Markov chain in Z only, where exact inference is tractable by the forward backward algorithm.

$$p(Z) \propto \int dx_1 p(z_1|x_1) \int dx_2 p(x_2|z_1)p(z_2|x_2) \int dx_3 p(x_3|z_2)p(z_3|x_3) \cdots \int dx_T p(x_T|z_{T-1})$$

(I've omitted the factors $p(y_t|x_t)$ here, for brevity. They may be considered absorbed into the respective factors $p(z_t|x_t)$.) The corresponding forward variables are shown here. These expressions, and similar ones for the backward variables, can be derived by induction on t.

$$\begin{aligned} A_1^{(z_1)} &= \int dx_1 p(x_1)p(z_1|x_1)p(y_1|x_1) \\ A_t^{(z_t)} &= \sum_{z_{t-1}}^{R_{t-1}} A_{t-1}^{(z_{t-1})} \int dx_t p(x_t|z_{t-1})p(z_t|x_t)p(y_t|x_t) \end{aligned}$$

But given Z, the x's are independent, so we can easily compute exact marginals or samples from $p(X)$ as well.

$$p(x_t|Y) \propto \sum_{z_{t-1}}^{R_{t-1}} A_{t-1}^{(z_{t-1})} \sum_{z_t}^{R_t} B_{t+1}^{(z_t)} p(x_t|z_{t-1})p(z_t|x_t)p(y_t|x_t)$$

The marginal distribution of x_t can be exactly expressed in terms of the neighboring forward backward variables of Z. So we're doing discrete forward backward to perform inference on continuous variables.

An important thing to note is that we can compute these exactly so long as the inner products here are evaluable:

$$\int dx_t p(x_t|z_{t-1})p(z_t|x_t)p(y_t|x_t)$$

These integrals correspond to the inner products of left and right singular vectors in the discrete Markov chain analogy, of which there are few kept from the "full rank" density. Also, note that we need only compute these inner product integrals once. They can be tabulated before inference begins.

If we can compute the inner product integrals, analytically or numerically, then inference takes $O(R^2T)$ time and $O(RT)$ storage (assuming a fixed $R_t = R$). If the potential function basis functions $p(z_t|x_t)$ and $p(x_{t+1}|z_t)$ are such that $p(x_{t+1}|x_t)$ is effectively banded, as would often be the case in smoothing applications, then the speed would scale linearly with the rank of the potentials as opposed to quadratically, further speeding up inference.

5.4 The von Mises distribution and smoothing potential

5.4.1 The von Mises distribution

The von Mises distribution is popular for modeling one-dimensional directional data, largely because the necessary normalization factors can be computed easily, and furthermore the density function is conjugate to itself, like the Gaussian. On the unit circle, the von Mises distribution can be parametrized by the "mean" angle ϕ where the mode of the density resides:

$$f(x|\phi,\kappa) = \frac{e^{\kappa\cos(x-\phi)}}{2\pi I_0(\kappa)}$$

where $I_0(x)$ is the order 0 modified Bessel function. (Assume throughout that κ is fixed.) Instead, however, consider the unit vector $\mathbf{x}$ from the circle's center to the point on the perimeter an angle x away from the x-axis. Similarly define μ to be the vector with tail at the circle's center and point on the perimeter making the angle ϕ with the x-axis. This we may write

$$f(\mathbf{x}|\mu,\kappa) = \frac{e^{\kappa\mu^T\mathbf{x}}}{2\pi I_0(\kappa)}.$$

This is more generally an expression for the p-dimensional von Mises-Fisher distribution on the surface of the p-sphere, where $\mathbf{x}$ and μ are p-vectors and where $(2\pi I_0(\kappa))^{-1}$ is replaced by the more general normalization constant

$$C_p(\kappa) = \frac{\kappa^{p/2-1}}{(2\pi)^{p/2} I_{p/2-1}(\kappa)}.$$

In particular,

$$C_3(\kappa) = \frac{\kappa}{4\pi \sinh \kappa}$$

A fairly simple algorithm for sampling from the von Mises distribution is described in [Best and Fisher, 1979] and implemented in the Circular Statistics Toolbox written for MATLAB by Peter Berens and described in [Berens, 2009]. Said toolbox also includes useful functions such as evaluation of the von Mises density function.

5.4.2 The von Mises smoothing prior

Suppose we have a time series of angular variables X that we know to vary gradually. Then we may want to model X with a low-rank chain distribution that couples neighboring x's in such a way both that smoothness is imposed on X and that the necessary inner products are very easy to compute. One such model is

$$p(X) \propto \prod_t \sum_{i=0}^{R} e^{\kappa_t \cos\left(x_t - \frac{2\pi i}{R+1}\right)} e^{\kappa_t \cos\left(x_{t+1} - \frac{2\pi i}{R+1}\right)} \tag{5.1}$$

For each time step t, this can be viewed as, and is, a superposition of unimodal functions of x_t and x_{t+1}, each peaked at $x_t = x_{t+1} = \frac{2\pi i}{R+1}$ for $i = 1, \cdots, R$, as illustrated in Figure 5.2.

5.5 Loopy orientation map model

On the cortical surface, we know that in many cases nearby columns of cells have similar tuning properties, so that if we want to estimate the tuning properties of some of these columns that we observe simultaneously then it makes sense to share information across neurons by smoothing. In primary visual cortex, the preferred orientation is usually a smooth function of position. See [Ohki et al., 2006] for an example of analysis of orientation preference data.

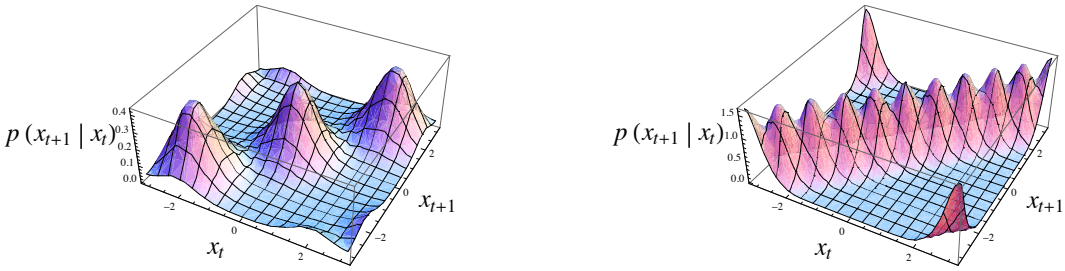


Figure 5.2: The von Mises smoother transition kernel, shown for rank $R = 3$ (left) and $R = 10$ (right), where $\kappa_t := R$ in each case.

5.5.1 Smooth prior and posterior distributions of an orientation map

Here we make a minor change in notation: Z is now observations, not auxiliary variables, which will go by k, l, and m. Figure 5.3 is a graphical model of the orientation preferences of an array of cortical columns. (The graphical model has rows and columns, and the nodes represent columns of cells. Hopefully it is clear from context which kind of column is being referred to in each instance. We will mostly refer to the columns of cells as *nodes*). Let $o_{i,j}$ denote the preferred orientation of the node at position (i, j). We make a noisy observation $z_{i,j}$ of each node's orientation preference. Then we have that

$$\begin{aligned} p(O, Z) &= p(O)p(Z|O) \\ p(Z|O) &= \prod_{i,j} p(z_{i,j}|o_{i,j}). \end{aligned} \tag{5.2}$$

We need a smoothing prior $p(O)$. One convenient choice is to use a low-rank spatial prior.

$$p(O) = p(o_{1,1}) \prod_{i,j} p(o_{i,j+1}|o_{i,j})p(o_{i+1,j}|o_{i,j}) \tag{5.3}$$

Here the term $p(o_{i,j+1}|o_{i,j})$ penalizes roughness in the horizontal direction, and $p(o_{i+1,j}|o_{i,j})$ penalizes the vertical direction. The following edge potentials serve to smooth orientation maps, coupling adjacent nodes to keep their values similar:

$$\begin{aligned} p(o_{i,j}|o_{i-1,j}) &\propto \sum_{k=0}^{R} e^{\kappa \cos\left(o_{i-1,j} - \frac{2\pi k}{R+1}\right)} e^{\kappa \cos\left(o_{i,j} - \frac{2\pi k}{R+1}\right)} \\ p(o_{i,j}|o_{i,j-1}) &\propto \sum_{k=0}^{R} e^{\kappa \cos\left(o_{i,j-1} - \frac{2\pi k}{R+1}\right)} e^{\kappa \cos\left(o_{i,j} - \frac{2\pi k}{R+1}\right)} \end{aligned} \tag{5.4}$$

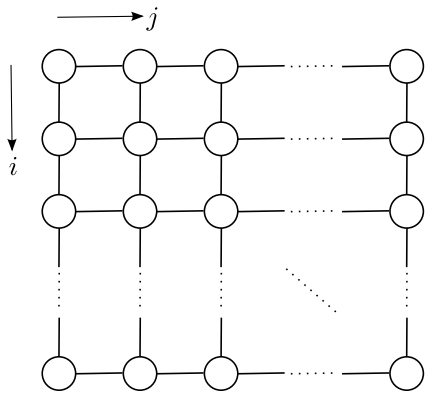


Figure 5.3: A graphical model of orientation selectivity, omitting observed nodes and their dependencies.

We can suppose that each node has a flat prior $p(o_{i,j})$, as we do here for simplicity, or we can suppose von Mises priors about set modes. Throughout, the concentration parameter κ is assumed to be fixed, though this assumption can be relaxed. With the observations included the graphical model may be depicted as in Figure 5.4, where the small black circles denote the observations which have probability

$$p(z_{i,j}|o_{i,j}) \propto e^{\kappa \cos(z_{i,j}-o_{i,j})}. \tag{5.5}$$

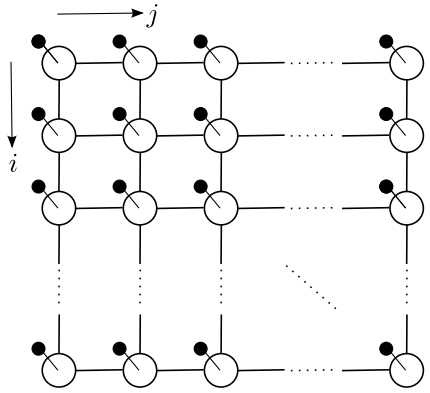


Figure 5.4: A graphical model of orientation selectivity, including observed nodes and their dependencies.

5.5.2 The conditional distribution of a column

Define $\mathbf{r}_k$ to be the unit vector that makes an angle of $\frac{2\pi k}{R+1}$ with the x-axis. Define $\mathbf{o}_{i,j}$ to be the vector that makes an angle of $o_{i,j}$ with the x-axis. Define $\mathbf{z}_{i,j}$ to be the unit vector that makes an angle of $z_{i,j}$ with the x-axis. Suppose that we have marginal estimates for every node and that we wish to block-wise draw a sample from the j^{th} column. Without loss of generality, assume $1 < j < N$ where N is the number of columns and M the number of rows. The conditional distribution of the j^{th} column given the rest of the nodes and given the observations is

$$\begin{aligned} P_j &\equiv p(O_{1:M,j}|O\backslash O_{1:M,j}, Z) \\ &= p(O_{1:M,j}|O_{1:M,j-1}, O_{1:M,j+1}, Z_{1:M,j}) \\ &= p(o_{1,j}|o_{1,j-1}, o_{1,j+1}, z_{1,j}) \prod_{i=2}^{M} p(o_{i,j}|o_{i-1,j}, o_{i,j-1}, o_{i,j+1}, z_{i,j}). \end{aligned} \tag{5.6}$$

This first factor is

$$\begin{aligned} p(o_{1,j}|o_{1,j-1}, o_{1,j+1}, z_{1,j}) &\propto p(o_{1,j-1})p(o_{1,j}|o_{1,j-1})p(o_{1,j+1}|o_{1,j})p(z_{1,j}|o_{1,j}) \\ &\propto p(o_{1,j}|o_{1,j-1})p(o_{1,j+1}|o_{1,j})p(z_{1,j}|o_{1,j}) \\ &\propto \sum_{k=0}^{R} e^{\kappa \mathbf{r}_k^T \mathbf{o}_{1,j-1}} e^{\kappa \mathbf{r}_k^T \mathbf{o}_{1,j}} \sum_{k'=0}^{R} e^{\kappa \mathbf{r}_{k'}^T \mathbf{o}_{1,j}} e^{\kappa \mathbf{r}_{k'}^T \mathbf{o}_{1,j+1}} \cdot e^{\kappa \mathbf{z}_{1,j}^T \mathbf{o}_{1,j}}. \end{aligned} \tag{5.7}$$

The second factor is

$$\begin{aligned} p(o_{i,j}|o_{i-1,j}, o_{i,j-1}, o_{i,j+1}, z_{i,j}) &\propto p(o_{i,j-1})p(o_{i-1,j})p(o_{i,j}|o_{i,j-1})p(o_{i,j+1}|o_{i,j})p(z_{i,j}|o_{i,j}) \\ &\propto p(o_{i,j}|o_{i,j-1})p(o_{i,j+1}|o_{i,j})p(z_{i,j}|o_{i,j}) \\ &\propto \sum_{k=0}^{R} e^{\kappa \mathbf{r}_k^T \mathbf{o}_{i,j-1}} e^{\kappa \mathbf{r}_k^T \mathbf{o}_{i,j}} \sum_{k'=0}^{R} e^{\kappa \mathbf{r}_{k'}^T \mathbf{o}_{i,j}} e^{\kappa \mathbf{r}_{k'}^T \mathbf{o}_{i,j+1}} \cdot e^{\kappa \mathbf{z}_{i,j}^T \mathbf{o}_{i,j}}. \end{aligned}$$

Putting these together, the conditional distribution is

$$\begin{aligned} P_j \propto \sum_{k=0}^{R} e^{\kappa \mathbf{r}_k^T \mathbf{o}_{1,j-1}} e^{\kappa \mathbf{r}_k^T \mathbf{o}_{1,j}} \sum_{k'=0}^{R} e^{\kappa \mathbf{r}_{k'}^T \mathbf{o}_{1,j}} e^{\kappa \mathbf{r}_{k'}^T \mathbf{o}_{1,j+1}} \cdot e^{\kappa \mathbf{z}_{1,j}^T \mathbf{o}_{1,j}} \times \\ \prod_{i=2}^{T} \sum_{k''=0}^{R} e^{\kappa \mathbf{r}_{k''}^T \mathbf{o}_{i,j-1}} e^{\kappa \mathbf{r}_{k''}^T \mathbf{o}_{i,j}} \sum_{k'''=0}^{R} e^{\kappa \mathbf{r}_{k'''}^T \mathbf{o}_{i,j}} e^{\kappa \mathbf{r}_{k'''}^T \mathbf{o}_{i,j+1}}. \end{aligned} \tag{5.8}$$

5.5.3 Sampling a column block-wise by the filter forward sample backward algorithm

Along column j, starting from the top ($i = 1$), we compute forward variables each corresponding to a particular value of the auxiliary value k_i that connects $o_{i,j}$ to $o_{i+1,j}$. It is helpful to have the joint factor graph representation in Figure 5.5 in mind. In this figure, the index of summation over an auxiliary variable labels the factor corresponding to that auxiliary variable.

$$
\begin{aligned}
A_1^{(k_1)} &= \int do_{1j} p(o_{1j}) p(k_1|o_{1j}) p(z_{1j}|o_{1j}) \\
&= \frac{1}{\mathcal{Z}_{A,1}} \int do_{1j} \left(\sum_{l_1=0}^{R} e^{\kappa \mathbf{r}_{l_1}^T \mathbf{o}_{1,j-1}} e^{\kappa \mathbf{r}_{l_1}^T \mathbf{o}_{1j}} \sum_{m_1=0}^{R} e^{\kappa \mathbf{r}_{m_1}^T \mathbf{o}_{1j}} e^{\kappa \mathbf{r}_{m_1}^T \mathbf{o}_{1,j+1}} \right) e^{\kappa \mathbf{r}_{k_1}^T \mathbf{o}_{1j}} e^{\kappa \mathbf{z}_{1j}^T \mathbf{o}_{1j}} \\
&= \frac{1}{\mathcal{Z}_{A,1}} \sum_{l_1=0}^{R} \sum_{m_1=0}^{R} e^{\kappa(\mathbf{r}_{l_1}^T \mathbf{o}_{1,j-1} + \mathbf{r}_{m_1}^T \mathbf{o}_{1,j+1})} \int do_{1j} e^{\kappa(\mathbf{r}_{l_1} + \mathbf{r}_{m_1} + \mathbf{r}_{k_1} + \mathbf{z}_{1j})^T \mathbf{o}_{1j}} \\
&= \frac{1}{\mathcal{Z}_{A,1}} \sum_{l_1=0}^{R} \sum_{m_1=0}^{R} e^{\kappa(\mathbf{r}_{l_1}^T \mathbf{o}_{1,j-1} + \mathbf{r}_{m_1}^T \mathbf{o}_{1,j+1})} \frac{1}{C_3(\kappa \cdot ||\mathbf{r}_{l_1} + \mathbf{r}_{m_1} + \mathbf{r}_{k_1} + \mathbf{z}_{1j}||)}
\end{aligned} \tag{5.9}
$$

$$
\begin{aligned}
A_i^{(k_i)} &= \sum_{k_{i-1}=0}^{R} A_{i-1}^{(k_{i-1})} \int do_{i,j} p(o_{i,j}|k_{i-1}) p(k_i|o_{i,j}) p(z_{i,j}|o_{i,j}) \\
&= \frac{1}{\mathcal{Z}_{A,i}} \sum_{k_{i-1}=0}^{R} A_{i-1}^{(k_{i-1})} \int do_{i,j} \left(\sum_{l_i=0}^{R} e^{\kappa \mathbf{r}_{l_i}^T \mathbf{o}_{i,j-1}} e^{\kappa \mathbf{r}_{l_i}^T \mathbf{o}_{i,j}} \sum_{m_i=0}^{R} e^{\kappa \mathbf{r}_{m_i}^T \mathbf{o}_{i,j}} e^{\kappa \mathbf{r}_{m_i}^T \mathbf{o}_{i,j+1}} \right) e^{\kappa \mathbf{r}_{k_{i-1}}^T \mathbf{o}_{i,j}} e^{\kappa \mathbf{r}_{k_i}^T \mathbf{o}_{i,j}} \\
&= \frac{1}{\mathcal{Z}_{A,i}} \sum_{k_{i-1}=0}^{R} A_{i-1}^{(k_{i-1})} \sum_{l_i=0}^{R} \sum_{m_i=0}^{R} e^{\kappa\left(\mathbf{r}_{l_i}^T \mathbf{o}_{i,j-1} + \mathbf{r}_{m_i}^T \mathbf{o}_{i,j+1}\right)} \int do_{i,j} e^{\kappa(\mathbf{r}_{l_i} + \mathbf{r}_{m_i} + \mathbf{r}_{k_{i-1}} + \mathbf{r}_{k_i} + \mathbf{z}_{i,j})^T \mathbf{o}_{i,j}} \\
&= \frac{1}{\mathcal{Z}_{A,i}} \sum_{k_{i-1}=0}^{R} A_{i-1}^{(k_{i-1})} \sum_{l_i=0}^{R} \sum_{m_i=0}^{R} e^{\kappa\left(\mathbf{r}_{l_i}^T \mathbf{o}_{i,j-1} + \mathbf{r}_{m_i}^T \mathbf{o}_{i,j+1}\right)} \frac{1}{C_3(\kappa \cdot ||\mathbf{r}_{l_i} + \mathbf{r}_{m_i} + \mathbf{r}_{k_{i-1}} + \mathbf{r}_{k_i} + \mathbf{z}_{i,j}||)}
\end{aligned} \tag{5.10}
$$

where the normalization constant for each time step t is computed by summing over the unnormalized variables $A_t^{(k_t)}$ with respect to k_t. With these variables in hand we can use the standard filter forward sample backward algorithm of hidden Markov model theory to generate samples from the prior (with observations set to zero) or posterior. Here's how:

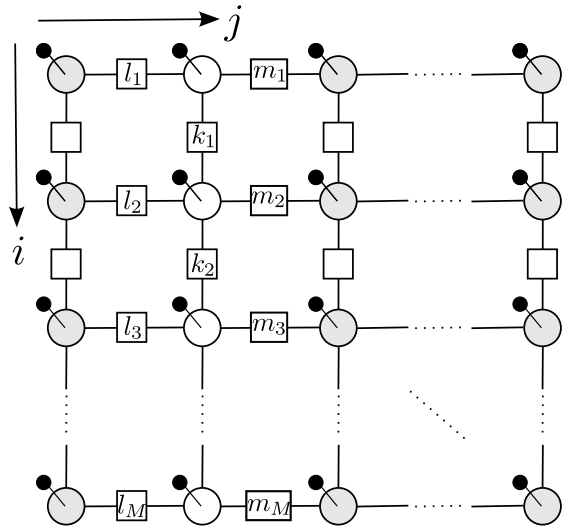


Figure 5.5: A graphical model of orientation selectivity, including observed nodes and auxiliary variables and their dependencies, drawing attention to a forward backward run on the second column from the left. The shaded nodes' values are held as constants, as are the observations. At each step of the forward recursion, to each possible value of k_i corresponds a forward variable $A_i^{(k_i)}$.

first we want to sample $o_{j,M}$ given $o_{j-1,M}$, $o_{j+1,M}$, and $z_{j,M}$. That marginal distribution is

$$
\begin{aligned}
&p(o_{M,j}|o_{M,j-1}, o_{M,j+1}, z_{M,j}) \\
&\quad = \sum_k A_{M-1}^{(k)} e^{\kappa \mathbf{r}_k^T \mathbf{o}_{M,j}} \sum_{k'=0}^{R} e^{\kappa \mathbf{r}_{k'}^T \mathbf{o}_{M,j-1}} e^{\kappa \mathbf{r}_{k'}^T \mathbf{o}_{M,j}} \sum_{k''=0}^{R} e^{\kappa \mathbf{r}_{k''}^T \mathbf{o}_{M,j+1}} e^{\kappa \mathbf{r}_{k''}^T \mathbf{o}_{M,j}} \cdot e^{\kappa \mathbf{z}_{M,j}^T \mathbf{o}_{M,j}} \\
&\quad = \sum_k A_{M-1}^{(k)} \sum_{k'=0}^{R} e^{\kappa \mathbf{r}_{k'}^T \mathbf{o}_{M,j-1}} \sum_{k''=0}^{R} e^{\kappa \mathbf{r}_{k''}^T \mathbf{o}_{M,j+1}} \left(e^{\kappa \mathbf{r}_k^T \mathbf{o}_{M,j}} e^{\kappa \mathbf{r}_{k'}^T \mathbf{o}_{M,j}} e^{\kappa \mathbf{r}_{k''}^T \mathbf{o}_{M,j}} e^{\kappa \mathbf{z}_{M,j}^T \mathbf{o}_{M,j}} \right)
\end{aligned}
$$

Next we compute the conditional distribution used to sample the rest of the elements of the column:

$$
\begin{aligned}
&p(o_{m,j}|o_{m+1,j}, o_{m,j-1}, o_{m,j+1}, z_{m,j}) \\
&\quad = \sum_k A^{(k)}_{m-1} \sum_{k'=0}^{R} e^{\kappa \mathbf{r}_{k'}^T \mathbf{o}_{m,j-1}} \sum_{k''=0}^{R} e^{\kappa \mathbf{r}_{k''}^T \mathbf{o}_{m,j+1}} \left(e^{\kappa \mathbf{r}_k^T \mathbf{o}_{m,j}} e^{\kappa \mathbf{r}_{k'}^T \mathbf{o}_{m,j}} e^{\kappa \mathbf{r}_{k''}^T \mathbf{o}_{m,j}} e^{\kappa \mathbf{z}_{m,j}^T \mathbf{o}_{m,j}} \right) \times \\
&\quad \sum_{k'''=0}^{R} e^{\kappa \mathbf{r}_{k'''}^T \mathbf{o}_{m+1,j}} e^{\kappa \mathbf{r}_{k'''}^T \mathbf{o}_{m,j}} \\
&\quad = \sum_k A^{(k)}_{m-1} \sum_{k'=0}^{R} e^{\kappa \mathbf{r}_{k'}^T \mathbf{o}_{m,j-1}} \sum_{k''=0}^{R} e^{\kappa \mathbf{r}_{k''}^T \mathbf{o}_{m,j+1}} \sum_{k'''=0}^{R} e^{\kappa \mathbf{r}_{k'''}^T \mathbf{o}_{m+1,j}} \times \\
&\quad \left(e^{\kappa \mathbf{r}_k^T \mathbf{o}_{m,j}} e^{\kappa \mathbf{r}_{k'}^T \mathbf{o}_{m,j}} e^{\kappa \mathbf{r}_{k''}^T \mathbf{o}_{m,j}} e^{\kappa \mathbf{r}_{k'''}^T \mathbf{o}_{m,j}} e^{\kappa \mathbf{z}_{m,j}^T \mathbf{o}_{m,j}} \right)
\end{aligned}
$$

5.5.4 Computing marginal distributions by the forward backward algorithm

As above with the forward variables we may derive backward variables:

$$
\begin{aligned}
B_M^{(k_{M-1})} &= \int do_{M,j} p(o_{M,j}|k_{M-1,j}) p(z_{M,j}|o_{M,j}) \\
&= \frac{1}{\mathcal{Z}_{B,M}} \sum_{l_M=0}^{R} \sum_{m_M=0}^{R} e^{\kappa \left(\mathbf{r}_{l_M}^T \mathbf{o}_{M,j-1} + \mathbf{r}_{m_M}^T \mathbf{o}_{M,j+1} \right)} \frac{1}{C_3(\kappa \cdot ||\mathbf{r}_{l_M} + \mathbf{r}_{m_M} + \mathbf{r}_{k_{M-1}} + \mathbf{z}_{M,j}||)}
\end{aligned} \tag{5.11}
$$

$$
\begin{aligned}
B_i^{k_{i-1}} &= \sum_{k_i=0}^{R} B_{i+1}^{k_i} \int do_{i,j} p(o_{i,j}|k_{i-1}) p(k_i|o_{i,j}) p(z_{i,j}|o_{i,j}) \\
&= \frac{1}{\mathcal{Z}_{B,i}} \sum_{k_i=0}^{R} B_{i+1}^{k_i} \sum_{l_i=0}^{R} \sum_{m_i=0}^{R} e^{\kappa \left(\mathbf{r}_{l_i}^T \mathbf{o}_{i,j-1} + \mathbf{r}_{m_i}^T \mathbf{o}_{i,j+1} \right)} \frac{1}{C_3(\kappa \cdot ||\mathbf{r}_{l_i} + \mathbf{r}_{m_i} + \mathbf{r}_{k_{i-1}} + \mathbf{r}_{k_i} + \mathbf{z}_{i,j}||)}
\end{aligned} \tag{5.12}
$$

With these in hand we are equipped to write down the marginal distribution of any given node:

$$p(o_{i,j}|Z) \propto \sum_{k_{i-1}=0}^{R} A_{i-1}^{(k_{i-1})} \sum_{k_i=0}^{R} B_{i+1}^{(k_i)} p(o_{i,j}|k_{i-1}) p(k_i|o_{i,j}) p(z_{i,j}|o_{i,j})$$
$$= \sum_{k_{i-1}=0}^{R} A_{i-1}^{(k_{i-1})} \sum_{k_i=0}^{R} B_{i+1}^{(k_i)} \left(\sum_{l_i=0}^{R} e^{\kappa \mathbf{r}_{l_i}^T \mathbf{o}_{i,j-1}} e^{\kappa \mathbf{r}_{l_i}^T \mathbf{o}_{i,j}} \sum_{m_i=0}^{R} e^{\kappa \mathbf{r}_{m_i}^T \mathbf{o}_{i,j}} e^{\kappa \mathbf{r}_{m_i}^T \mathbf{o}_{i,j+1}} \right) \times \tag{5.13}$$

$$e^{\kappa \mathbf{r}_{k_{i-1}}^T \mathbf{o}_{i,j}} e^{\kappa \mathbf{r}_{k_i}^T \mathbf{o}_{i,j}}.$$

5.5.5 Low-rank prior draw

Figure 5.6 is a sample from the loopy von Mises orientation preference model for rank $R = 5$.

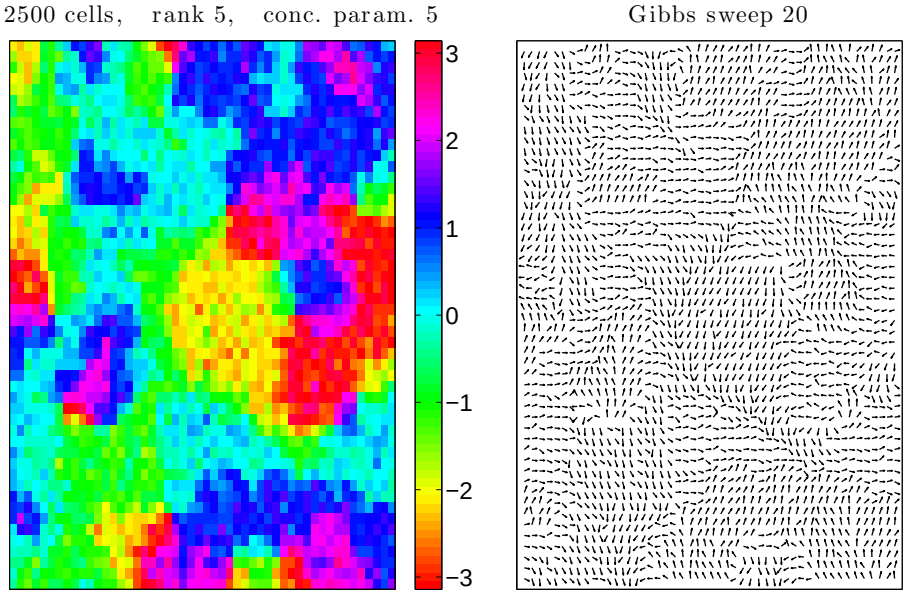


Figure 5.6: A sample 50×50 grid of angles drawn from the von Mises orientation map prior with rank $R = 5$ and $\kappa_t = 5 \ \forall t$. Twenty Gibbs sweeps were made.

5.6 A simpler phase selectivity map model

While the loopy low-rank model for orientation preference is tractable in principle – i.e. it has favorable scaling properties – we find that in absolute terms the corresponding Gibbs sampler is quite slow. The is especially so because computational cost scales quadratically for the rank, and for low rank there is not much smoothing. With this in mind, we consider a simpler model.

For our likelihood, we suppose that the true distribution over phases for any given spiking neuron i is von Mises with some mean vector $\vec{\mu}_i$ and concentration parameter $\vec{\kappa}_i$. As a prior, we assume that neuron i has a mean phase drawn from a mixture distribution parameterized by the mean phases of i's nearest neighbors and defined as follows:

$$p\left(\mu_i \left| \{\mu_j\}_{j\in N_i}\right.\right) \propto \prod_{j\in N_i} \left(a + b \cdot f(||\vec{x}_j - \vec{x}_i||) \cdot g(|\phi_j - \phi_i|)\right)$$
$$\equiv \prod_{j\in N_i} \left(0.005 + 0.04 \cdot \exp\left\{-\frac{1}{\tau}||\vec{x}_j - \vec{x}_i||\right\} \exp\left\{\kappa \cos(\phi_j - \phi_i)\right\} / e^{\kappa}\right)$$

Here κ is a global concentration parameter distinct from the observation concentration parameters $\{\kappa_i\}$ defined above. The $f(\cdot)$ function decreases with increasing distance between neurons, muting the influence of faraway neurons. The $g(\cdot)$ function is also unimodal (it is indeed a von Mises density). It promotes the closeness of the selectivities of physically nearby neurons. The additive constant makes the potential between each pair of neurons a spike-and-slab potential that allows for occassional sharp changes in the selectivity across the map. The particular parameter values were chosen to fit data described below.

5.7 Experimental data

We apply both the loopy low-rank model and the simpler model to the estimation of phase maps from subsampled observations of real mouse spinal neuron firing data.[1] In each case, we tune our model parameters using one data set, and then measure the performance of each tuned model by applying it to a second data set. The 3-dimensional spatial positions of the neurons have been precisely determined. We take the (vector) average of the full

[1] Phase selectivity data provided by Tim Machado.

set of observations for a particular neuron to be ground truth, and we subsample 5% of these observations to form the likelihood for each inference algorithm. According to this likelihood, each neuron's firing phase is a von Mises random variable with mean parameter equal to the average of the subsampled observations.

In the experimental setup, a mouse spinal cord is isolated in a dish and stimulated to evoke a network state similar to walking.[2] During walking, many muscles must be activated at precise times during the step cycle. The spinal cord can generate this pattern by itself while disconnected from muscles. It is known that motor neurons that control a particular neuron are spatially clustered together in the spinal cord; each such cluster is called a motor pool.

In the experiment, hundreds of motor neurons that connect to the mouse hindlimb were imaged during this walking-related network state. Obvious from the measurements (see the topmost panel of Figure 5.8 in Section 5.8) are spatial clusters of neurons with similar phase tuning. Presumably, the motor neurons that fire together are motor pools. If we can clearly identify which motor neurons belong to which pools then we can address a number of important questions: What is the correspondence between anatomical pools (which areas connect to a specific muscle) and functional pools? How may we explain the complexity of the network output during walking? Also, what is the variance in phase preference across a motor pool?

5.8 Phase map reconstruction: simpler model

We fix the concentration parameter $\kappa_i := 50$ for each neuron i. One might instead compute the maximum likelihood estimate of κ_i from the subsampled data for each neuron. For τ we choose the value 14. Inference is made tractable by exploiting the fact that the neuron-neuron potential falls off quickly with the distance between neurons. In our computations, the potential factors were only computed for the 10 nearest neighbors for each neuron.

The subsampled observation likelihood is the input to a Gibbs sampler that effectively smooths this selectivity map according to the simpler model prior. Results are shown in

[2]Much of the text in this section is adapted from communications with Tim Machado.

Figures 5.7 and 5.8. Here shown at 10,000 Gibbs sweeps, but even after only 1,000 or fewer sweeps, the Gibbs sampler yields an estimate that is more than 20% more accurate that the raw subsampled observation. We define the error to be proportional to the sum of the magnitudes of the vector differences between the true (fully observed) phase preference and the estimated preference. Figure 5.9 shows maps of the magnitude of the this vector error for the subsampled observation as well as for a Gibbs sample and the Gibbs running average.

There are two types of spinal neuron that are roughly 180 degrees out of phase with one another. This shows up in Figure 5.7 as the light blue horizontal bands near plus and minus pi. The presence of this feature makes inference with the simpler phase selectivity model less effective because this feature is not accounted for in the model. Notice that in the negative region of the vertical axis of the full observation in Figure 5.8 there are two regions (red and light blue) that are each fairly uniform within itself but are about 180 degrees out of phase with one another. This sharp boundary will not likely be recovered by inference with the simpler model. Rather, we expect that the two regions will bleed together, and this is indeed what we see in the Gibbs sample and average in the third and fourth panels of Figure 5.8. We have set the parameters of the simple model by hand, tuning the model to achieve the best possible reconstruction of the data. As a test of the resulting mode, we then fix these parameters and use the same model to reconstruct the selectivity map from a different set of subsampled data; see Figure 5.10. The smoothed estimate represents more than a 25% reduction in the error from the raw subsampled data, which is as good as the reduction in the case of the original data.

5.9 Phase map reconstruction: low-rank model

We applied a low-rank model as described above to analysis of the same spinal neuron data. First we formed a (loopy) graph by connecting each neuron with its two nearest neighbors. Connected neurons were coupled by the von Mises smoother potential in Equation (5.1). We set a global κ parameter to be equal to the rank R, which we set to 10. Increasing the rank only makes the smoother smooth more uniformly around the unit circle, so that larger values are best, but computational cost does increase quadratically with the rank.

A smoothed estimate of the selectivity map of the spinal neurons is shown in Figure 5.12. After 1,000 Gibbs sweeps, the sampler appears to have converged to the posterior average. It is clear that in this instance a simpler solution is preferable to the low-rank model, both for its accuracy and tractability.

As in the case of the simpler model, we tested our model with fixed parameters by using it to reconstruct a different dataset; see Figure 5.13. Once again, it is clear that the simple selectivity model outperforms the low-rank model, the latter reducing the error by less that 20%.

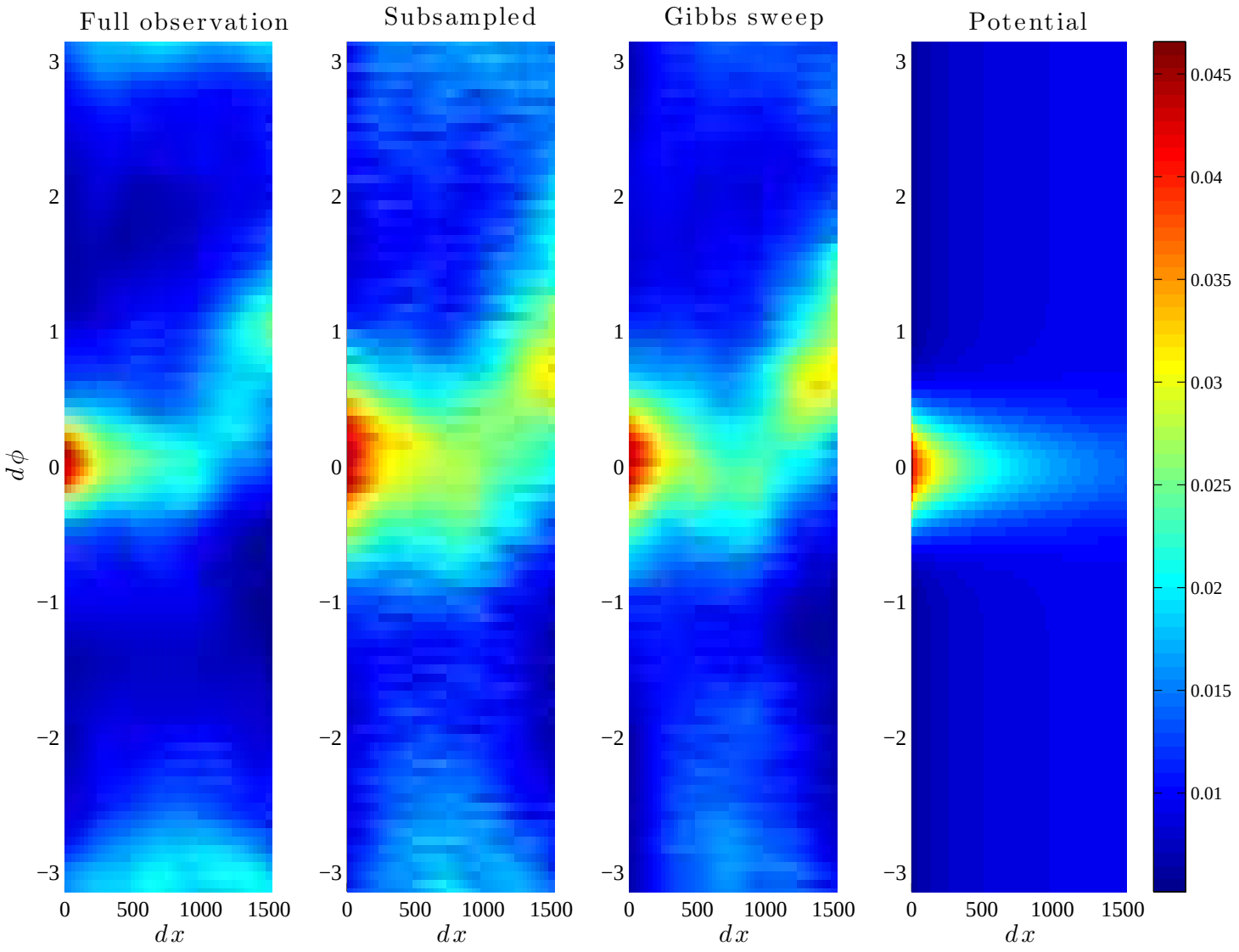

Full observation
Subsampled
Gibbs sweep
Potential
$d\phi$
dx
3
2
1
0
−1
−2
−3
0
500
1000
1500
0.045
0.04
0.035
0.03
0.025
0.02
0.015
0.01

Figure 5.7

Figure 5.7: Selectivity distributions and neuron-neuron potential for simple model. **Left:** The empirical distribution of the difference in phase preference between two neurons (vertical axis) in the full observation, for each (binned) value (horizontal axis) of the distance between two neurons. **Center:** The same empirical distribution as in the left panel but for the sample selectivity map at Gibbs sweep number 10,000. **Right:** The neuron-neuron potential, for each value of the distance between two neurons, as a function of the difference in their phase preferences.

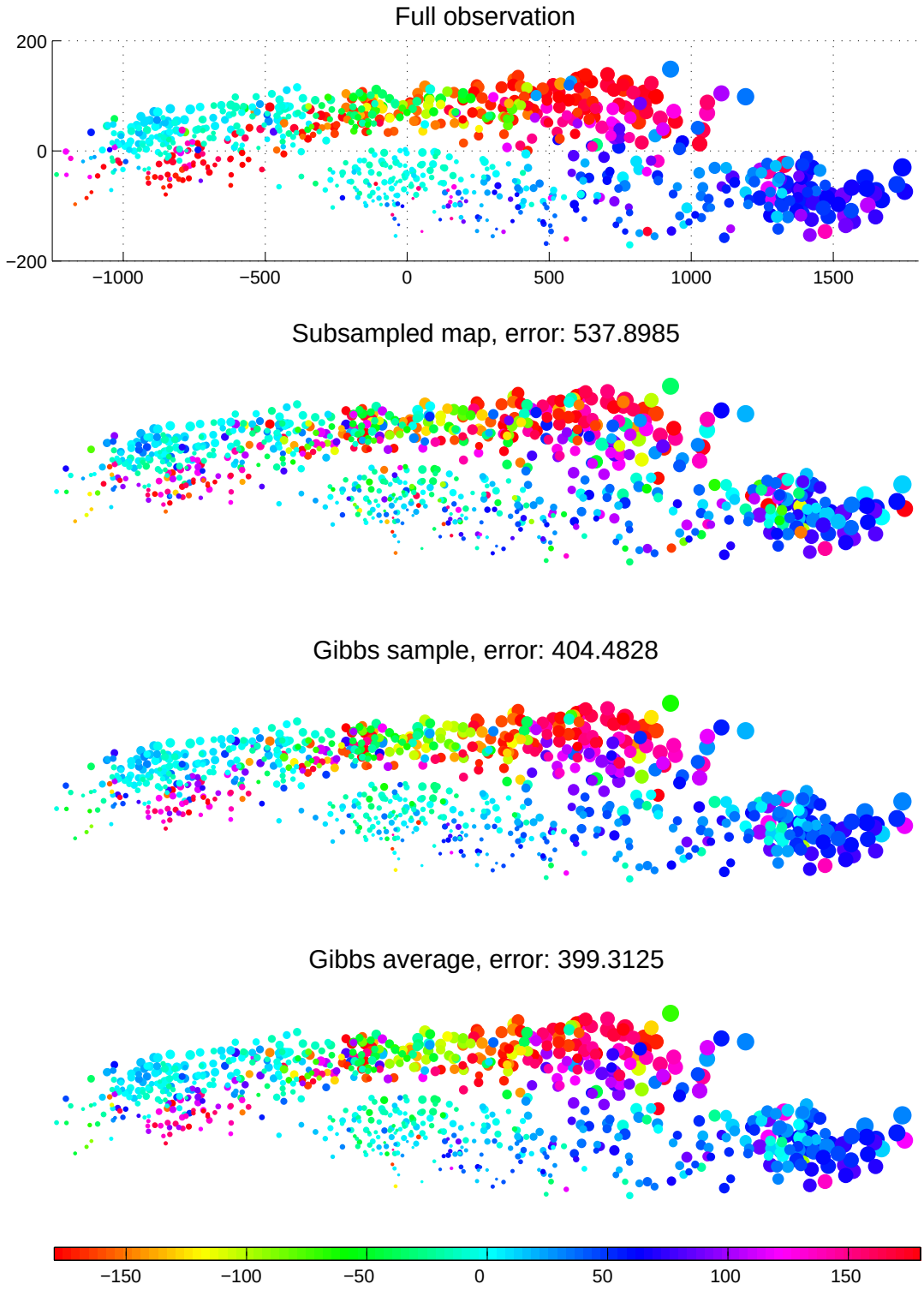

Full observation
200
0
−200
−1000
−500
0
500
1000
1500
Subsampled map, error: 537.8985
Gibbs sample, error: 404.4828
Gibbs average, error: 399.3125
−150
−100
−50
0
50
100
150

Figure 5.8

Figure 5.8: Progress of the Gibbs sampler for the simpler phase selectivity model applied to data from spinal neuron recordings. **First panel:** The full observation map. The third spatial dimension is represented by the varying size of the circles, each circle corresponding to a single neuron. The value of the phase preference is represented by the color of the circle, according to the colorbar at the far right. **Second panel:** The subsampled orientation selectivity map. **Third panel:** The selectivity map sample at Gibbs sweep number 10,000. **Fourth panel:** The selectivity map average over the first 10,000 Gibbs samples.

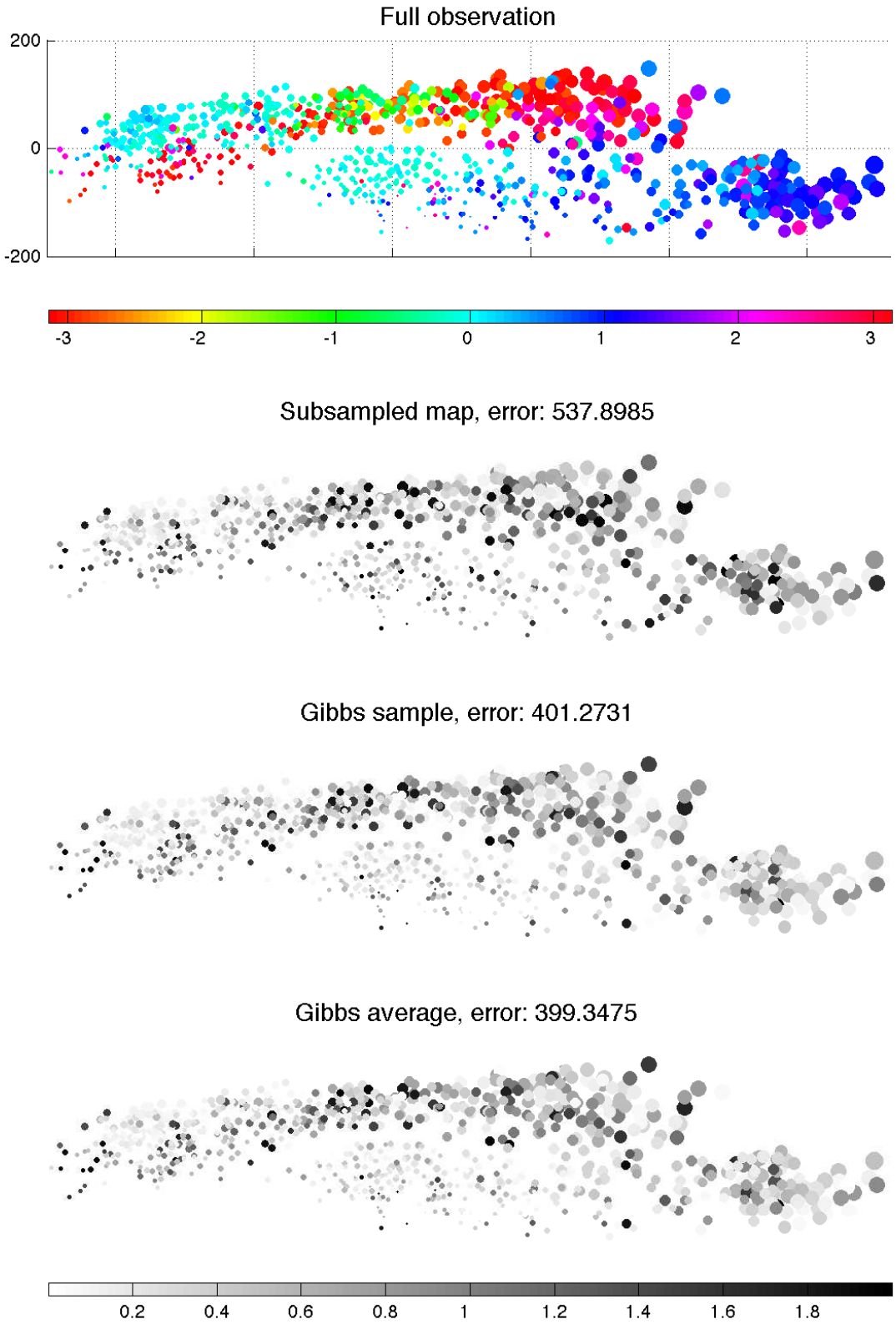

Full observation
200
0
-200
-3
-2
-1
0
1
2
3
Subsampled map, error: 537.8985
Gibbs sample, error: 401.2731
Gibbs average, error: 399.3475
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8

Figure 5.9

Figure 5.9: Maps of the errors in the estimate at different locations against that of the subsampled observation, for the simpler phase selectivity model applied to data from spinal neuron recordings. **First panel:** The full observation map. The third spatial dimension is represented by the varying size of the circles, each circle corresponding to a single neuron. The value of the phase preference is represented by the color of the circle, according to the colorbar at the far right. **Second panel:** The subsampled error map. Each point is colored corresponding to the vector difference of the subsampled and fully observed phase preferences. **Third panel:** The error map for Gibbs sweep number 10,000. **Fourth panel:** The error map for the average over the first 10,000 Gibbs samples.

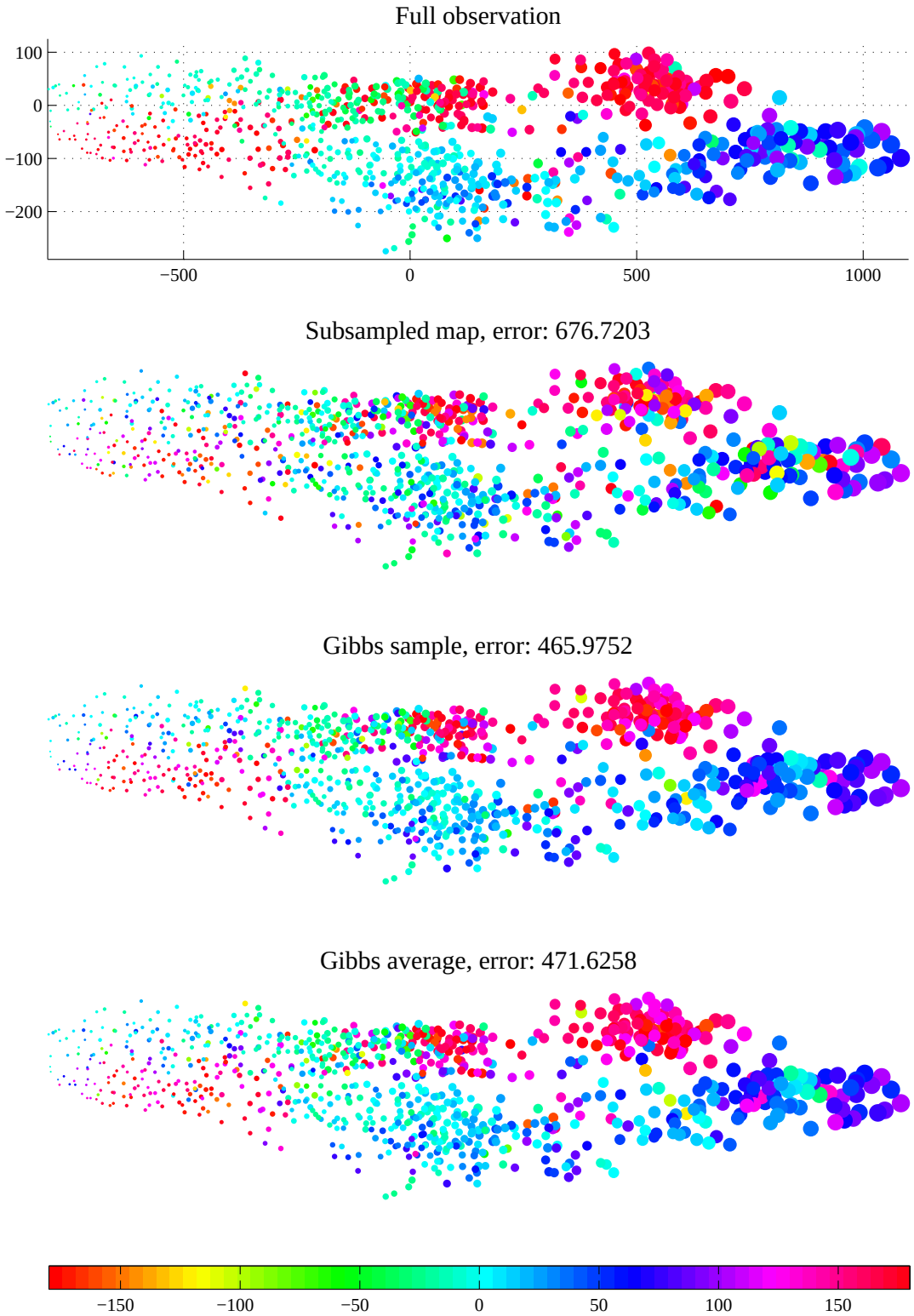

Full observation
100
0
−100
−200
−500
0
500
1000
Subsampled map, error: 676.7203
Gibbs sample, error: 465.9752
Gibbs average, error: 471.6258
−150
−100
−50
0
50
100
150

Figure 5.10

Figure 5.10: Progress of the Gibbs sampler for the simpler phase selectivity model applied to different data from spinal neuron recordings. **First panel:** The full observation map. The third spatial dimension is represented by the varying size of the circles, each circle corresponding to a single neuron. The value of the phase preference is represented by the color of the circle, according to the colorbar at the far right. **Second panel:** The subsampled orientation selectivity map. **Third panel:** The selectivity map sample at Gibbs sweep number 10,000. **Fourth panel:** The selectivity map average over the first 10,000 Gibbs samples.

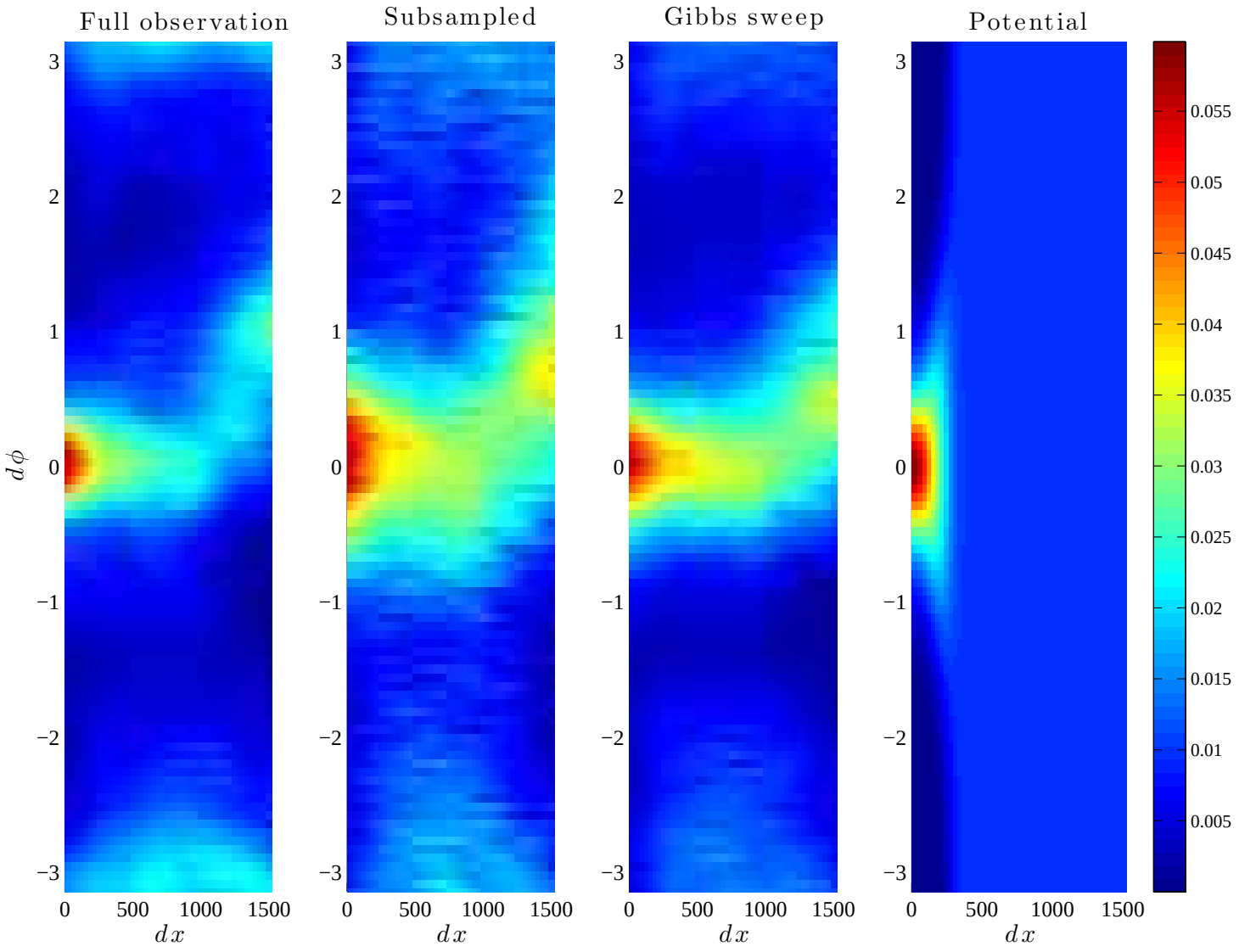

Full observation
Subsampled
Gibbs sweep
Potential
$d\phi$
dx
3
2
1
0
−1
−2
−3
0
500
1000
1500
0.055
0.05
0.045
0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005

Figure 5.11

Figure 5.11: Selectivity distributions and neuron-neuron potential for low-rank model. **Left:** The empirical distribution of the difference in phase preference between two neurons (vertical axis) in the full observation, for each (binned) value (horizontal axis) of the distance between two neurons. **Center:** The same empirical distribution as in the left panel but for the sample selectivity map at Gibbs sweep number 1,000. **Right:** The neuron-neuron potential, for each value of the distance between two neurons, as a function of the difference in their phase preferences.

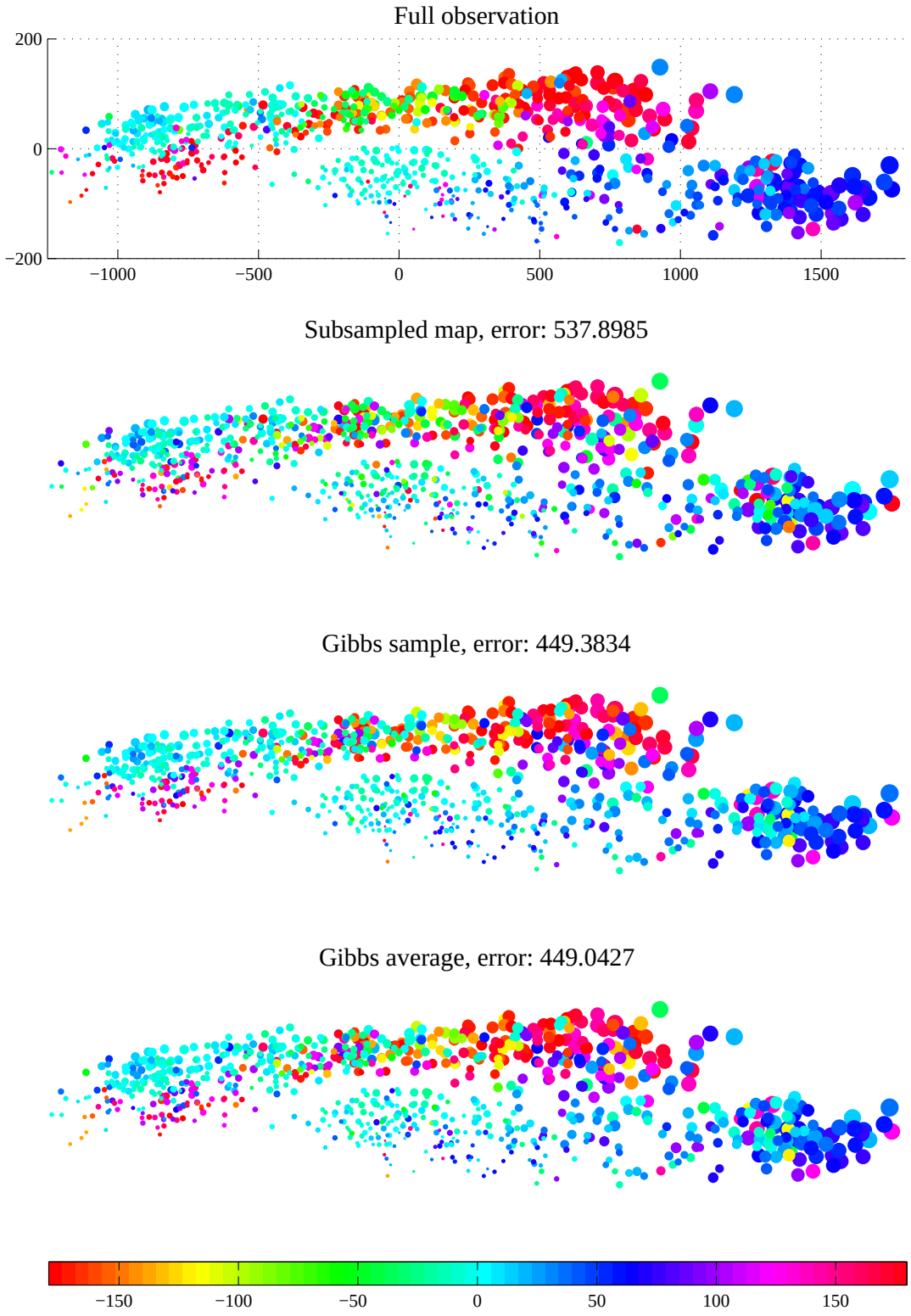

Full observation
200
0
−200
−1000
−500
0
500
1000
1500
Subsampled map, error: 537.8985
Gibbs sample, error: 449.3834
Gibbs average, error: 449.0427
−150
−100
−50
0
50
100
150

Figure 5.12

Figure 5.12: Progress of the Gibbs sampler for the low-rank phase selectivity model applied to data from spinal neuron recordings. **First panel:** The full observation map. The third spatial dimension is represented by the varying size of the circles, each circle corresponding to a single neuron. The value of the phase preference is represented by the color of the circle, according to the colorbar at the far right. **Second panel:** The subsampled orientation selectivity map. **Third panel:** The selectivity map sample at Gibbs sweep number 1,000. **Fourth panel:** The selectivity map average over the first 1,000 Gibbs samples.

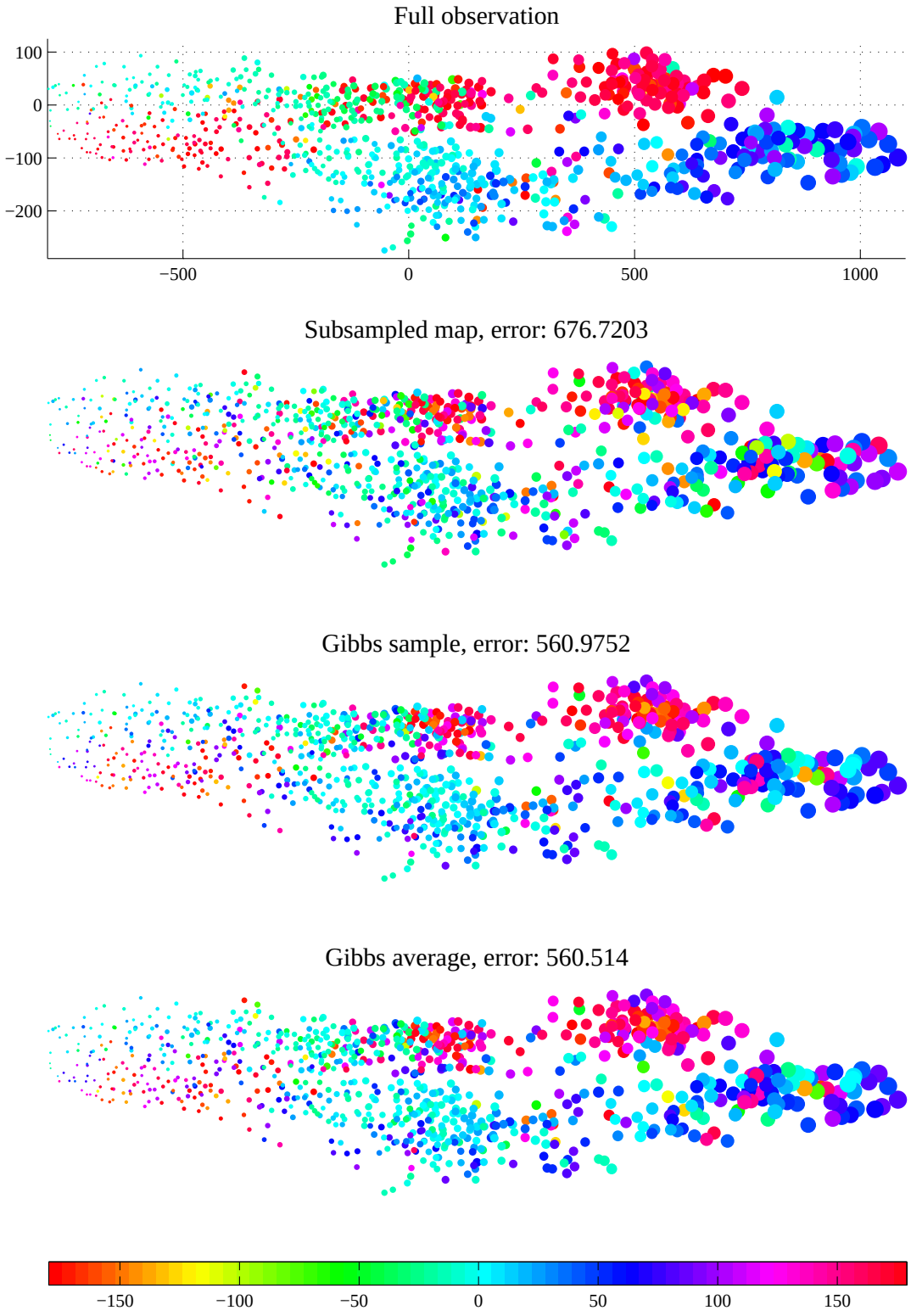

Full observation
100
0
−100
−200
−500
0
500
1000
Subsampled map, error: 676.7203
Gibbs sample, error: 560.9752
Gibbs average, error: 560.514
−150
−100
−50
0
50
100
150

Figure 5.13

Figure 5.13: Progress of the Gibbs sampler for the low-rank phase selectivity model applied to different data from spinal neuron recordings. **First panel:** The full observation map. The third spatial dimension is represented by the varying size of the circles, each circle corresponding to a single neuron. The value of the phase preference is represented by the color of the circle, according to the colorbar at the far right. **Second panel:** The subsampled orientation selectivity map. **Third panel:** The selectivity map sample at Gibbs sweep number 1,000. **Fourth panel:** The selectivity map average over the first 10,000 Gibbs samples.

Bibliography

Ahmadian, Y., Pillow, J., and Paninski, L. (2011). Efficient Markov Chain Monte Carlo methods for decoding population spike trains. *Neural Computation*, 23:46–96.

Aldworth, Z., Miller, J., Gedeon, T., Cummins, G., and Dimitrov, A. (2005). Dejittered spike-conditioned stimulus waveforms yield improved estimates of neuronal feature selectivity and spike-timing precision of sensory interneurons. *Journal of Neuroscience*, 25:5323–5332.

Amarasingham, A., Harrison, M. T., Hatsopoulos, N. G., and Geman, S. (2012). Conditional modeling and the jitter method of spike re-sampling. *Journal of Neurophysiology*, 107:517–531.

Berens, P. (2009). CircStat: A MATLAB toolbox for circular statistics. *Journal of Statistical Software*, 31(10):1–21.

Best, D. J. and Fisher, N. I. (1979). Efficient simulation of the von Mises distribution. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(2).

Bishop, C. (2006). *Pattern recognition and machine learning*. Springer.

Cadieu, C. F. and Koepsell, K. (2010). Phase coupling estimation from multivariate phase statistics. *Neural Computation*, 22(12):3107–3126.

Calabrese, A. and Paninski, L. (2011). Kalman filter mixture model for spike sorting of non-stationary data. *Journal of neuroscience methods*, 196:159–69.

Canepari, M., Djurisic, M., and Zecevic, D. (2007). Dendritic signals from rat hippocampal

CA1 pyramidal neurons during coincident pre- and post-synaptic activity: a combined voltage- and calcium-imaging study. *J Physiol*, 580(2):463–484.

Canepari, M., Vogt, K., and Zecevic, D. (2008). Combining voltage and calcium imaging from neuronal dendrites. *Cellular and Molecular Neurobiology*, 28:1079–1093.

Caron, F., Davy, M., and Doucet, A. (2007). Generalized Polya urn for time-varying Dirichlet process mixtures. In *Proc. 23rd Conf. on Uncertainty in Artificial Intelligence (UAI)*.

Carvalho, C. M., Polson, N. G., and Scott, J. G. (2010). The horseshoe estimator for sparse signals. *Biometrika*, 97(2):465–480.

Casella, G. and Berger, R. (2001). *Statistical Inference*. Duxbury Press.

Chen, Z., Kloosterman, F., Layton, S., and Wilson, M. A. (2012). Transductive neural decoding for unsorted neuronal spikes of rat hippocampus. In *Proc. IEEE EMBC '12, pp. 1310-1313, San Diego, CA*.

Chen, Z., Vijayan, S., Barbieri, R., Wilson, M. A., and Brown, E. N. (2009). Discrete- and continuous-time probabilistic models and algorithms for inferring neuronal up and down states. *Neural Comput.*, 21:1797–1862.

Cossart, R., Aronov, D., and Yuste, R. (2003). Attractor dynamics of network up states in the neocortex. *Nature*, 423:283–288.

Djurisic, M., Antic, S., Chen, W. R., and Zecevic, D. (2004). Voltage imaging from dendrites of mitral cells: EPSP attenuation and spike trigger zones. *J. Neurosci.*, 24(30):6703–6714.

Djurisic, M., Popovic, M., Carnevale, N., and Zecevic, D. (2008). Functional structure of the mitral cell dendritic tuft in the rat olfactory bulb. *J. Neurosci.*, 28(15):4057–4068.

Dombeck, D. A., Blanchard-Desce, M., and Webb, W. W. (2004). Optical recording of action potentials with second-harmonic generation microscopy. *J. Neurosci.*, 24(4):999–1003.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004a). Least angle regression. *Annals of Statistics*, 32(2):407–499.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004b). Least angle regression. *Annals of Statistics*, 32:407–499.

Escola, S., Fontanini, A., Katz, D., and Paninski, L. (2011). Hidden Markov models for the stimulus-response relationships of multistate neural systems. *Neural Computation*, pages 1–62.

Faisal, A. and Laughlin, S. (2007). Stochastic simulations on the reliability of action potential propagation in thin axons. *PLoS Comput Biol*, 3(5):e79.

Faisal, A. A., White, J. A., and Laughlin, S. B. (2005). Ion-channel noise places limits on the miniaturization of the brains wiring. *Curr Bio*, 15:1143–1149.

Fisher, J. A. N., Barchi, J. R., Welle, C. G., Kim, G.-H., Kosterin, P., Obaid, A. L., Yodh, A. G., Contreras, D., and Salzberg, B. M. (2008). Two-photon excitation of potentiometric probes enables optical recording of action potentials from mammalian nerve terminals in situ. *J Neurophysiol*, 99(3):1545–1553.

Fruhwirth-Schnatter, S. (2006). *Finite Mixture and Markov Switching Models*. Springer.

Gasthaus, J., Wood, F., Görür, D., and Teh, Y. (2009). Dependent Dirichlet process spike sorting. In *Advances in Neural Information Processing Systems*, pages 497–504.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2003). *Bayesian Data Analysis, Second Edition (Chapman & Hall/CRC Texts in Statistical Science)*. Chapman and Hall/CRC, 2 edition.

Gerstner, W. and Kistler, W. (2002). *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge University Press.

Goldwyn, J. H., Shea-Brown, E., and Rubinstein, J. T. (2010). Encoding and decoding amplitude-modulated cochlear implant stimuli - a point process analysis. *Journal of Computational Neuroscience*, 28(3):405–424.

Gollisch, T. (2006). Estimating receptive fields in the presence of spike-time jitter. *Network*, 17(2):103–129.

Goodman, I. and Johnson, D. (2008). Information theoretic bounds on neural prosthesis effectiveness: The importance of spike sorting. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 5204–5207.

Haslinger, R., Klinkner, K., and Shalizi, C. (2010). The computational structure of spike trains. *Neural computation*, 22:121–157.

Herbst, J. A., Gammeter, S., Ferrero, D., and Hahnloser, R. (2008). Spike sorting with hidden markov models. *Journal of Neuroscience Methods*, 174(1):126 – 134.

Hill, D. N., Mehta, S. B., and Kleinfeld, D. (2011). Quality metrics to accompany spike sorting of extracellular signals. *Journal of Neuroscience*, 31(24):8699–8705.

Horn, R. A. (1986). *Topics in matrix analysis*. Cambridge University Press, New York, NY, USA.

Huggins, J. and Paninski, L. (2012). Optimal experimental design for sampling voltage on dendritic trees. *In Press, Journal of Computational Neuroscience*.

Huys, Q., Ahrens, M., and Paninski, L. (2006). Efficient estimation of detailed single-neuron models. *Journal of Neurophysiology*, 96:872–890.

Huys, Q. and Paninski, L. (2009). Model-based smoothing of, and parameter estimation from, noisy biophysical recordings. *PLOS Computational Biology*, 5:e1000379.

Iyer, V., Hoogland, T. M., and Saggau, P. (2006). Fast functional imaging of single neurons using random-access multiphoton (RAMP) microscopy. *J Neurophysiol*, 95(1):535–545.

Kass, R. and Raftery, A. (1995). Bayes factors. *Journal of the American Statistical Association*, 90:773–795.

Kass, R., Ventura, V., and Brown, E. (2005). Statistical issues in the analysis of neuronal data. *Journal of neurophysiology*, 94:8–25.

Lewicki, M. (1998). A review of methods for spike sorting: the detection and classification of neural action potentials. *Network: Computation in Neural Systems*, 9:R53–R78.

Liu, J. S. (2008). *Monte Carlo Strategies in Scientific Computing*. Springer.

Mardia, K. and Jupp, P. (2000). *Directional statistics*. Wiley series in probability and statistics. Wiley.

Milojkovic, B. A., Zhou, W.-L., and Antic, S. D. (2007). Voltage and calcium transients in basal dendrites of the rat prefrontal cortex. *J Physiol*, 585(2):447–468.

Minka, T. (2001). *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, MIT.

Mishchenko, Y. and Paninski, L. (2011). Efficient methods for sampling spike trains in networks of coupled neurons. *Annals of Applied Statistics*, 5:1893–1919.

Mishchenko, Y., Vogelstein, J., and Paninski, L. (2011). A Bayesian approach for inferring neuronal connectivity from calcium fluorescent imaging data. *Annals of Applied Statistics*, 5(2B):1229–1261.

Mitchell, T. J. and Beauchamp, J. J. (1988). Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032.

Mohamed, S., Heller, K. A., and Ghahramani, Z. (2011). Bayesian and l1 approaches to sparse unsupervised learning. *CoRR*, abs/1106.1157.

Naud, R. and Gerstner, W. (2012). Coding and decoding with adapting neurons: A population approach to the peri-stimulus time histogram. *Plos Computational Biology*, 8(10).

Nossenson, N. and Messer, H. (2011). Optimal sequential detection of stimuli from multi-unit recordings taken in densely populated brain regions. *Neural Computation*, 24(4):895–938.

Nuriya, M., Jiang, J., Nemet, B., Eisenthal, K., and Yuste, R. (2006). Imaging membrane potential in dendritic spines. *PNAS*, 103:786–790.

Ohki, K., Chung, S., Ch'ng, Y., Kara, P., and Reid, C. (2005). Functional imaging with cellular resolution reveals precise micro-architecture in visual cortex. *Nature*, 433:597–603.

Ohki, K., Chung, S., Kara, P., Hubener, M., Bonhoeffer, T., and Reid, R. C. (2006). Highly ordered arrangement of single neurons in orientation pinwheels. *Nature*, 442(24):925–928.

Pakman, A., Huggins, J., and Paninski, L. (2012). Fast penalized state-space methods for inferring dendritic synaptic connectivity. *In preparation*.

Paninski, L. (2010). Fast Kalman filtering on quasilinear dendritic trees. *Journal of Computational Neuroscience*, 28:211–28.

Paninski, L. and Ferreira, D. (2008). State-space methods for inferring synaptic inputs and weights. *COSYNE*.

Paninski, L., Pillow, J., and Lewi, J. (2007). Statistical models for neural encoding, decoding, and optimal stimulus design. *Progress in brain research*, 165:493–507.

Park, T. and Casella, G. (2008). The Bayesian Lasso. *Journal of the American Statistical Association*, 103(482):681–686.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc.

Pillow, J., Ahmadian, Y., and Paninski, L. (2011). Model-based decoding, information estimation, and change-point detection techniques for multineuron spike trains. *Neural Computation*, 23(1):1–45.

Pitt, M. K. (2002). Constructing first order stationary autoregressive models via latent processes. *Scandinavian Journal of Statistics*, 29(4):657–663.

Pitt, M. K. and Walker, S. G. (2005). Constructing stationary time series models using auxiliary variables with applications. *Journal of the American Statistical Association*, 100(470):554–564.

Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2).

Reddy, G. D. and Saggau, P. (2005). Fast three-dimensional laser scanning scheme using acousto-optic deflectors. *J Biomed Opt*, 10(6):064038.

Robert, C. P. and Casella, G. (2005). *Monte Carlo Statistical Methods (Springer Texts in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Sacconi, L., Dombeck, D. A., and Webb, W. W. (2006). Overcoming photodamage in second-harmonic generation microscopy: Real-time optical recording of neuronal action potentials. *Proceedings of the National Academy of Sciences*, 103(9):3124–3129.

Sahani, M. (1999). *Latent variable models for neural data analysis*. PhD thesis, California Institute of Technology.

Siddiqi, S., Boots, B., and Gordon, G. (2010). Reduced-rank hidden markov models. *In Proc. 13th Intl. Conf. on Artificial Intelligence and Statistics (AISTATS)*.

Sjostrom, P. J., Rancz, E. A., Roth, A., and Hausser, M. (2008). Dendritic Excitability and Synaptic Plasticity. *Physiol. Rev.*, 88(2):769–840.

Smith, C., Wood, F., and Paninski, L. (2012). Low rank continuous-space graphical models. In *Proc. 15th Intl. Conf. on Artificial Intelligence and Statistics (AISTATS)*, pages 1064–1072.

Song, L., Gretton, A., and Guestrin, C. (2010a). Nonparametric tree graphical models via kernel embeddings. In *Proc. 13th Intl. Conf. on Artificial Intelligence and Statistics (AISTATS)*.

Song, L., Siddiqi, S. M., Gordon, G. J., and Smola, A. J. (2010b). Hilbert space embeddings of hidden markov models. In *International Conference on Machine Learning*, pages 991–998.

Tokdar, S., Xi, P., Kelly, R. C., and Kass, R. E. (2010). Detection of bursts in extracellular spike trains using hidden semi-markov point process models. *J. Comput. Neurosci.*, 29:203–212.

Toyoizumi, T., Rad, K., and Paninski, L. (2009). Mean-field approximations for coupled populations of generalized linear model spiking neurons with Markov refractoriness. *Neural computation*, 21(5):1203–1243.

Truccolo, W., Eden, U., Fellows, M., Donoghue, J., and Brown, E. (2005). A point process framework for relating neural spiking activity to spiking history, neural ensemble and extrinsic covariate effects. *Journal of Neurophysiology*, 93:1074–1089.

Turner, R. E. and Sahani, M. (2011). Probabilistic amplitude and frequency demodulation. In *Advances in Neural Information Processing Systems*, pages 981–989.

Ventura, V. (2008a). Automatic spike sorting using tuning information. *Submitted*.

Ventura, V. (2008b). Spike train decoding without spike sorting. *Neural computation*, 20(4):923–963.

Vidne, M., Ahmadian, Y., Shlens, J., Pillow, J., Kulkarni, J., Litke, A., Chichilnisky, E., Simoncelli, E., and Paninski, L. (2012). Modeling the impact of common noise inputs on the network activity of retinal ganglion cells. *J Comput Neurosci*, 33:97–121.

Vucinic, D. and Sejnowski, T. J. (2007). A compact multiphoton 3d imaging system for recording fast neuronal activity. *PLoS ONE*, 2(8):e699.

Wainwright, M. and Jordan, M. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305.

Wood, F. and Black, M. (2008). A nonparametric Bayesian alternative to spike sorting. *Journal of Neuroscience Methods*, 173:1–12.