ARTICLE **Communicated by Emery Brown**

Efficient Markov Chain Monte Carlo Methods for Decoding Neural Spike Trains

Yashar Ahmadian
yashar@stat.columbia.edu
Department of Statistics and Center for Theoretical Neuroscience, Columbia University, New York, New York 10027, U.S.A.

Jonathan W. Pillow
pillow@mail.utexas.edu
Center for Perceptual Systems, University of Texas at Austin, Austin, TX 78751, U.S.A.

Liam Paninski
liam@stat.columbia.edu
Department of Statistics and Center for Theoretical Neuroscience, Columbia University, New York, New York 10027, U.S.A.

Stimulus reconstruction or decoding methods provide an important tool for understanding how sensory and motor information is represented in neural activity. We discuss Bayesian decoding methods based on an encoding generalized linear model (GLM) that accurately describes how stimuli are transformed into the spike trains of a group of neurons. The form of the GLM likelihood ensures that the posterior distribution over the stimuli that caused an observed set of spike trains is log concave so long as the prior is. This allows the maximum a posteriori (MAP) stimulus estimate to be obtained using efficient optimization algorithms. Unfortunately, the MAP estimate can have a relatively large average error when the posterior is highly nongaussian. Here we compare several Markov chain Monte Carlo (MCMC) algorithms that allow for the calculation of general Bayesian estimators involving posterior expectations (conditional on model parameters). An efficient version of the hybrid Monte Carlo (HMC) algorithm was significantly superior to other MCMC methods for gaussian priors. When the prior distribution has sharp edges and corners, on the other hand, the "hit-and-run" algorithm performed better than other MCMC methods. Using these algorithms, we show that for this latter class of priors, the posterior mean estimate can have a considerably lower average error than MAP, whereas for gaussian priors, the two estimators have roughly equal efficiency. We also address the application of MCMC methods for extracting nonmarginal properties of the posterior distribution. For example, by using MCMC to calculate the mutual

© 2010 Massachusetts Institute of Technology

information between the stimulus and response, we verify the validity of a computationally efficient Laplace approximation to this quantity for gaussian priors in a wide range of model parameters; this makes direct model-based computation of the mutual information tractable even in the case of large observed neural populations, where methods based on binning the spike train fail. Finally, we consider the effect of uncertainty in the GLM parameters on the posterior estimators.

1 Introduction

Understanding the exact nature of the neural code is a central goal of theoretical neuroscience. Neural decoding provides an important method for comparing the fidelity and robustness of different codes (Rieke, Warland, de Ruyter van Steveninck, & Bialek, 1997). The decoding problem, in its general form, is the problem of estimating the relevant stimulus, $\mathbf{x}$, that elicited the observed spike trains, $\boldsymbol{r}$, of a population of neurons over a course of time. Neural decoding is also of crucial importance in the design of neural prosthetic devices (Donoghue, 2002).

A large literature exists on developing and applying different decoding methods to spike train data, in both single cell and population decoding. Bayesian methods lie at the basis of a major group of these decoding algorithms (Sanger, 1994; Zhang, Ginzburg, McNaughton, & Sejnowski, 1998; Brown, Frank, Tang, Quirk, & Wilson, 1998; Maynard et al., 1999; Stanley & Boloori, 2001; Shoham et al., 2005; Barbieri et al., 2004; Wu et al., 2004; Brockwell, Rojas, & Kass, 2004; Kelly & Lee, 2004; Karmeier, Krapp, & Egelhaaf, 2005; Truccolo, Eden, Fellows, Donoghue, & Brown, 2005; Jacobs, Grzywacz, & Nirenberg, 2006; Yu et al., 2009; Gerwinn, Macke, & Bethge, 2009; see also the companion article in this issue: J. Pillow, Y. Ahmadian, & L. Paninski, "Model-based decoding, information estimation, and change-point detection in multineuron spike trains," referred to hereafter as the companion article, Pillow et al., 2011). In such methods, the a priori distribution of the sensory signal, $p(\mathbf{x})$, is combined, via Bayes' rule, with an encoding model describing the probability, $p(\boldsymbol{r} \mid \mathbf{x})$, of different spike trains given the signal, to yield the posterior distribution, $p(\mathbf{x} \mid \boldsymbol{r})$, that carries all the information contained in the observed spike train responses about the stimulus. A Bayesian estimate is one that, given a definite cost function on the amount of error, minimizes the expected error cost under the posterior distribution. Assuming the prior distribution and the encoding model are appropriately chosen, the Bayes estimate is thus optimal by construction. Furthermore, since the Bayesian approach yields a distribution over the possible stimuli that could lead to the observed response, Bayes estimates naturally come equipped with measures of their reliability or posterior uncertainty.

In a fully Bayesian approach, one has to be able to evaluate any desired functional of the high-dimensional posterior distribution.

Unfortunately, calculating these can be computationally very expensive. For example, most Bayesian estimates involve integrations over the (often very high-dimensional) space of possible signals. Accordingly, most work on Bayesian decoding of spike trains has focused on cases where the signal is low dimensional (Sanger, 1994; Maynard et al., 1999; Abbott & Dayan, 1999; Karmeier et al., 2005) or on situations where the joint distribution, $p(\mathbf{x}, \boldsymbol{r})$, has a certain Markov tree decomposition, so that computationally efficient recursive techniques may be applied (Zhang et al., 1998; Brown et al., 1998; Barbieri et al., 2004; Wu et al., 2004; Brockwell et al., 2004; Kelly & Lee, 2004; Shoham et al., 2005; Eden, Frank, Barbieri, Solo, & Brown, 2004; Truccolo et al., 2005; Ergun, Barbieri, Eden, Wilson, & Brown, 2007; Yu et al., 2009; Paninski et al., 2010). The Markov setting is extremely useful; it lends itself naturally to many problems of interest in neuroscience and has thus been fruitfully exploited. In particular, this setting is very useful in an important class of decoding problems where stimulus estimation is performed online: the stimulus at some time, t, is estimated conditioned on the observation of the spike trains only up to that time, as opposed to the entire spike train.

Some decoding problems cannot be formulated in the online estimation framework. In such cases quantities of interest should naturally be conditioned on the entire history of the spike train. In this article, we focus on this latter class of problems (although many of the methods we discuss can potentially be adopted to the online case as well). Furthermore, it is awkward to cast many decoding problems of interest in the Markov setting. A more general method that does not require such tree decomposition properties is to calculate the maximum a posteriori (MAP) estimate $\mathbf{x}_{\text{MAP}}$ (Stanley & Boloori, 2001; Jacobs et al., 2006; Gerwinn et al., 2009; see the companion paper, Pillow et al., 2011, for further review and discussion). The MAP estimate requires no integration, only maximization of the posterior distribution, and can remain computationally tractable even when the stimulus space is very high dimensional. This is the case for general log-concave posterior distributions; many problems in sensory and motor coding fall in this class (it should be noted, however, that in many cases of interest where this condition is not satisfied, for example, when the distributions are inherently multimodal, posterior maximization can become highly intractable). The MAP is a good estimator when the posterior is well approximated by a gaussian distribution centered at $\mathbf{x}_{\text{MAP}}$ (Tierney & Kadane, 1986; Kass, Tierney, & Raftery, 1991). Because the mode and the mean of a gaussian distribution are identical, the MAP in this case is approximately equal to the posterior mean as well. This gaussian approximation is expected to be sufficiently accurate, for example, when the prior distribution and the likelihood function (i.e., $p(\boldsymbol{r} \mid \mathbf{x})$ as function of $\mathbf{x}$) are not very far from gaussian or when the likelihood is sharply concentrated around $\mathbf{x}_{\text{MAP}}$. However, when the prior distribution has sharp boundaries and corners and the likelihood function does not constrain the estimate away from such nongaussian regions, the gaussian approximation can fail, resulting in a large average error in the MAP estimate. In such cases, one expects the MAP to be inferior to the posterior mean $E(\mathbf{x} \mid \boldsymbol{r})$, which is the optimal estimate under squared error loss.

Accordingly, in section 3 of this article we develop efficient Markov chain Monte Carlo (MCMC) techniques for sampling from general log-concave posterior distributions and compare their performance in situations relevant to our neural decoding setting (for comprehensive introductions to MCMC methods, including their application in Bayesian problems, see, e.g., Robert & Casella, 2005 and Gelman, 2004). By providing a tool for approximating averages (integrals) over the exact posterior distribution, $p(\mathbf{x} \mid \boldsymbol{r}, \theta)$ (where θ are the parameters of the encoding forward model, in principle obtained by fitting to experimental data), these techniques allow us to calculate general Bayesian estimates such as $E(\mathbf{x} \mid \boldsymbol{r}, \theta)$ and provide estimates of their uncertainty. Although in principle many of the MCMC methods we discuss are applicable even to posterior distributions that are not log concave, they may lose their efficiency in such cases, and estimates based on them may not even converge to true posterior averages. In section 4 we compare the MAP and the posterior mean stimulus estimates based on the simulated response of a population of retinal ganglion cells (RGC). In section 5 we discuss the applications of MCMC for calculating more complicated properties of $p(\mathbf{x} \mid \boldsymbol{r}, \theta)$ beyond marginal statistics, such as the statistics of first-passage times. We also discuss an MCMC-based method known as bridge sampling (Bennett, 1976; Meng & Wong, 1996) that provides a tool for a direct calculation of the mutual information. Using this technique, we show that for gaussian priors, the estimates of Pillow et al. (2011) in the companion article for this quantity based on the Laplace approximation are robust and accurate. Finally, in section 6 we discuss the effect of uncertainty in the parameters of the forward model, θ, on the MAP and posterior mean estimate. We proceed by first introducing the forward model used to calculate the likelihood $p(\boldsymbol{r} \mid \mathbf{x}, \theta)$.

2 The Encoding Model, the MAP, and the Stimulus Ensembles

In this section, we give an overview of neural encoding models based on generalized linear models (GLM) (Brillinger, 1988; McCullagh & Nelder, 1989; Paninski, 2004; Truccolo et al., 2005), and briefly review the treatment of Pillow et al. (2011) in the companion article for MAP-based decoding. (Note that much of the material in this section is covered in this companion article, but we include a brief review here to make this article self-contained.) A neural encoding model assigns a conditional probability to the neural response given the stimulus. We take the stimulus to be an artificially discretized, possibly multicomponent, function of time, $x(t, n)$, which will be represented as a d-dimensional vector $\mathbf{x}$.[1]

In response to $\mathbf{x}$, the ith neuron emits a spike train response,

$$r_i(t) = \sum_{\alpha} \delta(t - t_{i,\alpha}), \tag{2.1}$$

where $t_{i,\alpha}$ is the time of the αth spike of the ith neuron. We represent this function by $\boldsymbol{r}_i$ (we use boldface symbols for both continuous time and discretized, finite-dimensional vectors) and the collection of response data of all cells by $\boldsymbol{r}$.

The response, $\boldsymbol{r}$, is not fully determined by $\mathbf{x}$ and is subject to trial-to-trial variations. We model $\boldsymbol{r}$ as a point process whose instantaneous firing rate is the output of a GLM (Brillinger, 1988; McCullagh & Nelder, 1989; Paninski, 2004). This class of models has been extensively discussed in the literature. Briefly, it is a generalization of the popular linear-nonlinear-Poisson model that includes feedback and interaction between neurons, with parameters that have natural neurophysiological interpretations (Simoncelli, Paninski, Pillow, & Schwartz, 2004) and has been applied in a wide variety of experimental settings (Brillinger, 1992; Dayan & Abbott, 2001; Chichilnisky, 2001; Theunissen et al., 2001; Brown, Barbieri, Eden, & Frank, 2003; Paninski, Fellows, Shoham, Hatsopoulos, & Donoghue, 2004; Truccolo et al., 2005; Pillow et al., 2008). The model gives the conditional (on the stimulus, as well as the history of the observed spike train) instantaneous firing rate of the ith observed cell as

$$\lambda_i(t) \equiv f\left(b_i + \sum_{\tau,n} k_i(t-\tau, n)x(\tau, n) + \sum_j \sum_\beta h_{ij}(t - t_{j,\beta})\right), \tag{2.2}$$

which we write more concisely as

$$\boldsymbol{\lambda}_i = f\left(b_i + K_i \cdot \mathbf{x} + \sum_j \mathcal{H}_{ij} \cdot \boldsymbol{r}_j\right). \tag{2.3}$$

Here, the linear operators (filters) K_i and $\mathcal{H}_{ij}$ have causal,[2] time-translation-invariant kernels $k_i(t, n)$ and $h_{ij}(t)$ (we note that the causality condition for $k_i(t, n)$ is true only for sensory neurons). The kernel $k_i(t, n)$ represents the ith cell's linear receptive field, and $h_{ij}(t)$ describe possible excitatory or

---

[1] The dimension of $\mathbf{x}$ is thus $d = NT$, where T is the number of time steps and N is the total number of components at each time step.

[2] That is, the kernels $k_i(t, n)$ and $h_{ij}(t)$ vanish for $t < 0$.

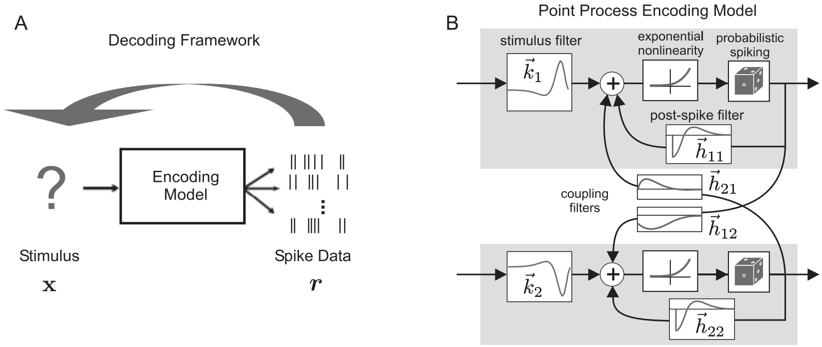
A
Decoding Framework
?
Encoding Model
Stimulus
$\mathbf{x}$
Spike Data
$\boldsymbol{r}$
B
Point Process Encoding Model
stimulus filter
exponential nonlinearity
probabilistic spiking
$\vec{k}_1$
post-spike filter
$\vec{h}_{11}$
coupling filters
$\vec{h}_{21}$
$\vec{h}_{12}$
$\vec{k}_2$
$\vec{h}_{22}$

Figure 1: Illustration of Bayesian decoding paradigm. (A) Bayesian decoding performs inference about the stimulus using the observed spike times and a specified encoding model. (B) Schematic of the encoding model (generalized linear model) used for the decoding examples shown in this article. The model parameters (k_i and h_{ij}) can be easily fit using maximum likelihood. Once fit, the model provides a description of the data likelihood, $p(\boldsymbol{r} \mid \mathbf{x})$, which is combined with the prior $p(\mathbf{x})$ to estimate $\mathbf{x}$.

inhibitory postspike effects of the jth observed neuron on the ith. The diagonal components h_{ii} describe the postspike feedback of the neuron to itself, and can account for refractoriness, adaptation and burstiness depending on their shape (see Paninski, 2004, for details). The constant b_i is the DC bias of the ith cell, such that $f(b_i)$ may be considered as the ith cell's constant baseline firing rate. Finally, $f(\cdot)$ is a nonlinear, nonnegative, increasing function.[3] Figure 1B shows a schematic diagram of the GLM encoding model for two cells.

Given the firing rate, equation 2.3, the forward probability, $p(\boldsymbol{r} \mid \mathbf{x}, \theta)$, can be written as (Snyder & Miller, 1991; Paninski, 2004; Truccolo et al., 2005)

$$\begin{aligned}\log p(\boldsymbol{r} \mid \mathbf{x}, \theta) &= \sum_i \left[\boldsymbol{r}_i^{\mathrm{T}} \log \boldsymbol{\lambda}_i - \int_0^T \lambda_i(t)\, dt \right] + const.\\ &= \sum_{i,\alpha} \log \lambda(t_{i,\alpha}) - \sum_i \int_0^T \lambda_i(t)\, dt + const., \end{aligned} \tag{2.4}$$

where $\theta = \{b_i, k_i, h_{ij}\}$ is the set of GLM parameters. The constant term serves to normalize the probability and does not depend on $\mathbf{x}$ or θ. We will restrict ourselves to $f(u)$ that are convex and log concave (e.g., this is the case for $f(u) = \exp(u)$). Then the log-likelihood function $L(\mathbf{x}, \theta)$ is guaranteed to be a separately concave function of either the stimulus $\mathbf{x}$ or the model parameters,[4] regardless of the observed spike data $\boldsymbol{r}$. The log concavity with respect to the model parameters makes maximum likelihood fitting of this model very easy, as concave functions on convex parameter spaces have no nonglobal local maxima. Therefore, simple gradient ascent algorithms can be used to find the maximum likelihood estimate.

---

[3] We note that even though the nonlinearity, $f(\cdot)$, has to be an increasing function, with appropriately chosen negative postspike feedback filters, h_{ii}, the mean firing rate of the GLM modeled neurons can still exhibit saturation as a function of the input strength, $\mathbf{x}$.

The prior distribution describes the statistics of the stimulus in the natural world or that of an artificial stimulus ensemble used by the experimentalist. In this article, we consider only priors relevant for the latter case. Given a prior distribution, $p(\mathbf{x})$, and having observed the spike trains, $\boldsymbol{r}$, the posterior probability distribution over the stimulus is given by Bayes' rule,

$$p(\mathbf{x} \mid \boldsymbol{r}, \theta) = \frac{p(\boldsymbol{r} \mid \mathbf{x}, \theta) p(\mathbf{x})}{p(\boldsymbol{r} \mid \theta)}, \tag{2.5}$$

where

$$p(\boldsymbol{r} \mid \theta) = \int p(\boldsymbol{r} \mid \mathbf{x}, \theta) p(\mathbf{x})\, d\mathbf{x}. \tag{2.6}$$

The MAP estimate is, by definition,

$$\begin{aligned} \mathbf{x}_{\text{MAP}}(\boldsymbol{r}, \theta) &= \arg\max_{\mathbf{x}} p(\mathbf{x} \mid \boldsymbol{r}, \theta) \\ &= \arg\max_{\mathbf{x}} \left[\log p(\boldsymbol{r} \mid \mathbf{x}, \theta) + \log p(\mathbf{x})\right], \end{aligned} \tag{2.7}$$

(Except in section 6, we will drop θ from the arguments of $\mathbf{x}_{\text{MAP}}$ or the distributions, it being understood that they are conditioned on the specific θ obtained from the experimental fit.) As discussed above, for the GLM nonlinearities that we consider, the likelihood, $p(\boldsymbol{r} \mid \mathbf{x}, \theta)$, is log concave in $\mathbf{x}$. If the prior, $p(\mathbf{x})$, is also log concave, then the posterior distribution is log concave as a function of $\mathbf{x}$, and its maximization (see equation 2.7) can also be achieved using simple gradient ascent techniques. The class of log-concave prior distributions is quite large, and it includes exponential, triangular, and general gaussian distributions, as well as uniform distributions with convex support.[5]

---

[4] That is, for fixed θ, it is a concave function of $\mathbf{x}$, and vice versa, but in general not a concave function of $(\mathbf{x}, \theta)$ jointly.

[5] Let us mention, however, that no first principle dictates that the posterior distribution over a biologically or behaviorally relevant variable (e.g., an external variable that a part of the brain seeks to estimate) should be log concave. In fact, distributions that are not log concave, such as multimodal or very fat-tailed distributions, can be highly relevant in biological settings.

The MAP is a good low-error estimate when Laplace's method provides a good approximation for the posterior mean, which has the minimum mean square error. This method is a general asymptotic method for approximating integrals when the integrand peaks sharply at its global maximum and is exponentially suppressed away from it. In the Bayesian setting, this corresponds to posterior integrals of interest (e.g., posterior averages and so-called Bayes factors) receiving their dominant contribution from the vicinity of the main mode of $p(\mathbf{x} \mid \boldsymbol{r}, \theta)$, that is, $\mathbf{x}_{\text{MAP}}$ (for a comprehensive review of Laplace's method in Bayesian applications, see Kass et al., 1991, and books on Bayesian analysis, such as Berger, 1993). In that case, we can Taylor expand the log posterior to the first nonvanishing order around $\mathbf{x}_{\text{MAP}}$ (i.e., the second order, since the derivative vanishes at the maximum), obtaining the gaussian approximation (hereafter also referred to as the Laplace approximation)

$$p(\mathbf{x} \mid \boldsymbol{r}, \theta) \approx e^{-\frac{1}{2}(\mathbf{x}-\mathbf{x}_{\text{MAP}})^{\text{T}} J\,(\mathbf{x}-\mathbf{x}_{\text{MAP}})+const.}. \tag{2.8}$$

Here the matrix J is the Hessian of the negative log posterior at $\mathbf{x}_{\text{MAP}}$,

$$J \equiv J_{ab}(\boldsymbol{r}, \theta) = -\frac{\partial^2 \log p(\mathbf{x} \mid \boldsymbol{r}, \theta)}{\partial x_a\, \partial x_b}\bigg|_{\mathbf{x}=\mathbf{x}_{\text{MAP}}}. \tag{2.9}$$

Normally in the statistical setting, the Laplace approximation is formally justified in the limit of large samples due to the central limit theorem, leading to a likelihood function with a very sharp peak (in neural decoding, the meaning of "large samples" depends in general on the nature of the stimulus, a point we discuss further in section 4). However, this approximation often proves adequate even for moderately strong likelihoods as long as the posterior is not grossly nonnormal. An obvious case where the approximation fails is for strongly multimodal distributions where no particular mode dominates. Here, we restrict our attention to the class of log-concave posteriors that are unimodal. For this class and for a smooth enough GLM nonlinearity, $f(\cdot)$, we expect equation 2.8 to hold for prior distributions that are close to normal, even when the likelihood is not extremely sharp. However, for flatter priors with sharp boundaries or "corners" we expect it to fail unless the likelihood is narrowly concentrated away from such nongaussian regions.

In this article, we set out to verify this intuition by studying two extreme cases within the class of log-concave priors, gaussian and flat distributions with convex support, given by

$$p(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^d \mid \mathcal{C} \mid}} e^{-\frac{1}{2}\mathbf{x}^{\text{T}}\mathcal{C}^{-1}\mathbf{x}} \tag{2.10}$$

and

$$p(\mathbf{x}) \propto I_{\mathcal{S}}(\mathbf{x}), \tag{2.11}$$

respectively.[6] Here, $\mathcal{C}$ is the $d \times d$ covariance matrix, and $I_{\mathcal{S}}$ is the indicator function of a convex region, $\mathcal{S}$, of $\mathbf{R}^d$. In particular, for the white-noise stimuli we consider in section 4, $\mathcal{C} = c^2\mathbf{I}_{d\times d}$ in the gaussian case, and $\mathcal{S}$ is the d-dimensional cube $[-\sqrt{3}c, \sqrt{3}c]^d$ in the flat case (this choice of $\mathcal{S}$ corresponds to a uniformly distributed white noise stimulus). Here, c is the standard deviation of the stimulus on a subinterval, and in the case where $x(t)$ is the normalized light intensity (with the average luminosity removed), it is referred to as the contrast. We compare the performance of the MAP and posterior mean estimates in each case in section 4. In section 5.2 we verify the adequacy of this approximation for the estimation of the mutual information in the case of gaussian priors.

3 Monte Carlo Techniques for Bayesian Estimates

For completeness, we start this section by reviewing the basics of the Markov chain Monte Carlo (MCMC) method (for comprehensive textbooks on MCMC methods, see, e.g., Gelman, 2004 and Robert & Casella, 2005). However, the main point of this section is the discussion of the applications of this method to the neural case and ways of making the method more efficient, as well as a comparison of the efficiency of different MCMC algorithms, in this specific setting. As noted in section 1, the posterior distribution, equation 2.5, represents the full information about the stimulus as encoded in the prior distribution and carried by the observed spike trains, $\boldsymbol{r}$. However, a much simpler (and therefore less complete) representation of this information can be provided by a so-called Bayesian estimate for the stimulus, possibly accompanied by a corresponding estimate of its error. A commonly used Bayesian estimate is the posterior mean,

$$E(\mathbf{x} \mid \boldsymbol{r}) = \int \mathbf{x}\, p(\mathbf{x} \mid \boldsymbol{r})\, d\mathbf{x}, \tag{3.1}$$

---

[6] White or correlated gaussian priors are often used in neural applications (e.g., gaussian stimuli are widely used in neurophysiological experiments). Flat priors with infinitely sharp boundaries are less biologically motivated. However, flat priors are the best log-concave approximation to binary priors, which are also quite common in sensory physiology as both binary white noise and M-sequences (Pillow et al., 2008; Reid, Victor, & Shapley, 1997). In this article, we consider the flat prior mainly as a limiting case of concave log priors with sharp derivatives when we check the accuracy of the Laplace approximation and compare the efficiency of various MCMC chains (see section 3.5) in different regimes.

which is the optimal estimator with respect to average square error. The uncertainty of this estimator is in turn provided by the posterior covariance matrix. When the posterior distribution can be reasonably approximated as gaussian, the posterior mean can be approximated by its mode, that is, the MAP estimate, equation 2.7, and the inverse of the log-posterior Hessian, equation 2.9, can represent its uncertainty. We adopt here the posterior mean, $E(\mathbf{x} \mid \boldsymbol{r})$, as a benchmark for comparing the performance of the two estimates, and we take the deviation of the MAP from the latter as a measure of the validity of the gaussian approximation for the posterior distribution.

To calculate the posterior mean, equation 3.1, we have to perform a high-dimensional integral over $\mathbf{x}$. Computationally this is quite costly. The Monte Carlo method is based on the idea that if one could generate N independent and identically distributed (i.i.d.) samples, $\mathbf{x}_t$ $(t = 1, \ldots, N)$, from a probability distribution, $\pi(\mathbf{x})$,[7] then one could approximate integrals involved in expectations such as equation 3.1 by sample averages. This is because by the law of large numbers, for any $g(\mathbf{x})$ (such that $\int \mid g(\mathbf{x}) \mid \pi(\mathbf{x})\, d\mathbf{x} < \infty$),

$$\hat{g}_N^{(\pi)} \equiv \frac{1}{N}\sum_{t=1}^{N} g(\mathbf{x}_t) \longrightarrow E_\pi(g) = \int g(\mathbf{x})\pi(\mathbf{x})\, d\mathbf{x}, \quad \text{as } N \to \infty. \tag{3.2}$$

Also, to decide how many samples are sufficient, we may estimate

$$\mathrm{Var}(\hat{g}_N) = \frac{1}{N}\mathrm{Var}[g(\mathbf{x})]; \tag{3.3}$$

when N is large enough that this variance is sufficiently small, we may stop sampling. However, it is often quite challenging to sample directly from a complex multidimensional distribution, and the efficiency of methods yielding i.i.d. samples often decreases exponentially with the number of dimensions.

Fortunately, equation 3.2 (the law of large numbers) still holds if the i.i.d. samples are replaced by an ergodic Markov chain, $\mathbf{x}_t$, whose equilibrium distribution is $\pi(\mathbf{x})$. This is the idea behind the MCMC method based on the Metropolis-Hastings (MH) algorithm (Metropolis, Rosenbluth, Rosenbluth, Teller, & Teller, 1953; Hastings, 1970). In the general form of this algorithm, the Markov transitions are constructed as follows. Starting at point $\mathbf{x}$, we first sample a point $\mathbf{y}$ from some "proposal" density $q(\mathbf{y} \mid \mathbf{x})$ and then accept this point as the next point in the chain, with probability

---

[7] We are, of course, interested in calculating posterior expectations corresponding to the case $\pi(\mathbf{x}) = p(\mathbf{x} \mid \boldsymbol{r})$, but as the discussion is general, we use $\pi(\mathbf{x})$ in the rest of this section for ease of notation.

$$\alpha(\mathbf{y} \mid \mathbf{x}) \equiv \min\left(1, \frac{\pi(\mathbf{y})q(\mathbf{x} \mid \mathbf{y})}{\pi(\mathbf{x})q(\mathbf{y} \mid \mathbf{x})}\right). \tag{3.4}$$

If $\mathbf{y}$ is rejected, the chain stays at point $\mathbf{x}$, so that the conditional Markov transition probability, $T(\mathbf{y} \mid \mathbf{x})$, is given by

$$T(\mathbf{y} \mid \mathbf{x}) = \alpha(\mathbf{y} \mid \mathbf{x})q(\mathbf{y} \mid \mathbf{x}) + R(\mathbf{x})\delta(\mathbf{y} - \mathbf{x}), \tag{3.5}$$

where

$$R(\mathbf{x}) = 1 - \int \alpha(\mathbf{y} \mid \mathbf{x})q(\mathbf{y} \mid \mathbf{x})\, d\mathbf{y}, \tag{3.6}$$

is the rejection probability of proposals from $\mathbf{x}$. The reason for accepting the proposals according to equation 3.4 is that doing so guarantees that $\pi(\mathbf{x})$ is invariant under the Markov evolution (see, e.g., Robert & Casella, 2005, for details). It is important to note that from equation 3.4, to execute this algorithm we need only to know $\pi(\mathbf{x})$ up to a constant, which is an advantage because often, particularly in Bayesian settings, normalizing the distribution itself requires the difficult integration for which we are using MCMC (we discuss a method of calculating the normalization constant in section 5.2).

The major drawback of the MCMC method is that the generated samples are dependent, and thus it is harder to estimate how long we need to run the chain to get an accurate estimate, and in general we may need to run the chain much longer than the i.i.d. case. Thus, we would like to choose a proposal density, $q(\mathbf{y} \mid \mathbf{x})$, that gives rise to a chain that explores the support of $\pi(\mathbf{x})$ (i.e., mixes) quickly, and has a small correlation time (roughly the number of steps separation to yield i.i.d samples), to reduce the number of steps the chain has to be iterated and hence the computational time (see section 3.5 and Gelman, 2004, and Robert & Casella, 2005, for further details). In general, a good proposal density $q(\mathbf{y} \mid \mathbf{x})$ should allow for large jumps with higher probability for falling in regions of larger $\pi(\mathbf{x})$ (so as to avoid a high MH rejection rate). A good rule of thumb is for the proposals $q(. \mid \mathbf{x})$ to resemble the true density $\pi(.)$ as much as possible. We review a few useful well-known proposals and explore different ways of boosting their efficiency in the GLM-based neural decoding setting. We note here that these algorithms can be applied to general distributions and do not require the log-concavity condition for $\pi(\mathbf{x})$. However, some of the enhancements that we consider can only be implemented, or are only expected to boost up the performance of the chain, when the distribution $\pi(\mathbf{x})$ is log concave (see the discussion of nonisotropic proposals in sections 3.1 and 3.2 and that of adaptive rejection sampling in section 3.4).

3.1 Nonisotropic Random Walk Metropolis (RWM).

Perhaps the most common proposal is of the random walk type: $q(\mathbf{x} \mid \mathbf{y}) = q(\mathbf{x} - \mathbf{y})$, for some fixed density $q(.)$. Centered isotropic gaussian distributions are a simple choice, leading to proposals

$$\mathbf{y} \sim \mathbf{x} + \sigma \mathbf{z}, \tag{3.7}$$

where $\mathbf{z}$ is gaussian of zero mean and identity covariance, and σ determines the proposal jump scale. (In this simple form, the RWM chain was used in a recent study to fit a hierarchical model of tuning curves of neurons in the primary visual cortex to experimental data: Cronin, Stevenson, Sur, & Kording, 2009.) Of course, different choices of the proposal distribution will affect the mixing rate of the chain. To increase this rate, it is generally a good idea to align the axes of $q(.)$ with the target density, if possible, so that the proposal jump scales in different directions are roughly proportional to the width of $\pi(\mathbf{x})$ along those directions. Such proposals will reduce the rejection probability and increase the average jump size by biasing the chain to jump in more favorable directions. For gaussian proposals, we can thus choose the covariance matrix of $q(.)$ to be proportional to the covariance of $\pi(\mathbf{x})$. Of course, calculating the latter covariance is often a difficult problem (which the MCMC method is intended to solve), but we can exploit the Laplace approximation, equation 2.8, and take the inverse of the Hessian of the log posterior at MAP, equation 2.9, as a first approximation for the covariance. This is equivalent to modifying the proposal rule 3.7 into

$$\mathbf{y} \sim \mathbf{x} + \sigma A \mathbf{z}, \tag{3.8}$$

where A is the Cholesky decomposition of J^{-1}

$$AA^{\mathrm{T}} = J^{-1}, \tag{3.9}$$

and J was defined in equation 2.9. We refer to chains with such jump proposals as nonisotropic gaussian RWM. Figure 2 compares the isotropic and nonisotropic proposals. The modification, equation 3.8, is equivalent to running a chain with isotropic proposals, equation 3.7, but for the auxiliary distribution $\tilde{\pi}(\tilde{\mathbf{x}}) =\mid A \mid \pi(A\tilde{\mathbf{x}})$ (whose corresponding Laplace approximation corresponds to a standard gaussian with identity covariance), and subsequently transforming the samples, $\tilde{\mathbf{x}}_t$, by the matrix A to obtain samples $\mathbf{x}_t = A\tilde{\mathbf{x}}_t$ from $\pi(\mathbf{x})$. Implementing nonisotropic sampling using the transformed distribution $\tilde{\pi}(\tilde{\mathbf{x}})$, instead of modifying the proposals as in equation 3.8 is more readily extended to chains more complicated than RWM

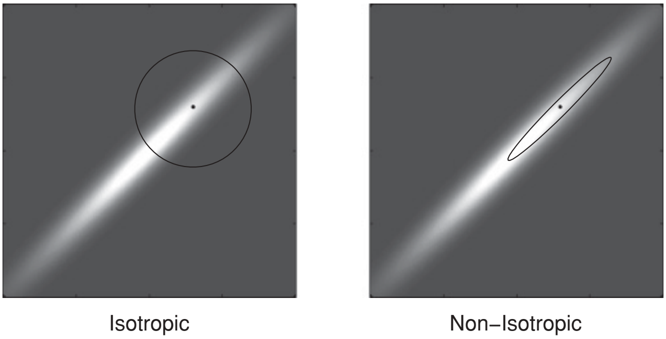

Figure 2: Comparison of isotropic and nonisotropic Markov jumps for the gaussian RWM and hit-and-run chains. In the RWM case, the circle and the ellipse are level sets of the gaussian proposal distributions for jumping from the dot at their center. In isotropic (nonisotropic) hit-and-run, the jump direction $\mathbf{n}$ is generated by normalizing a vector sampled from an isotropic (nonisotropic) gaussian distribution centered at the origin. The nonisotropic distributions were constructed using the Hessian, equation 2.9, in the Laplace approximation, so that the ellipse is described by $\mathbf{x}^{\mathrm{T}} J\, \mathbf{x} = const$. When the underlying distribution, $\pi(\mathbf{x})$, is highly nonisotropic, it is disadvantageous to jump isotropically, as it reduces the average jump size and slows the chain. In RWM, the proposal jump scale cannot be much larger than the scale of the narrow waist of the underlying distribution, lest the rejection rate gets large (as most proposals will fall in the dark region of small $\pi(\mathbf{x})$) and the chain gets stuck. For hit-and-run, there is no jump scale to be set by the user, and the jump size in a given direction, $\mathbf{n}$, is set by the scale of the "slice" distribution, equation 3.13. Thus, in the isotropic case, the average jump size will effectively be a uniform average over the scales of $\pi(\mathbf{x})$ along its principal axes. In the nonisotropic case, however, the jump size will be determined mainly by the scale of the "longer" dimensions, as the nonisotropic distribution gives more weight to these.

(see below), and therefore we used this latter method in our simulations using different chains.

As we will see in the next section, in the flat prior case and for weak stimulus filters or a small number of identical cells, the Laplace approximation can be poor. In particular, the Hessian, equation 2.9, does not contain any information about the prior in the flat case, and therefore the approximate distribution, equation 2.8, can be significantly broader than the extent of the prior support in some directions. To take advantage of the Laplace approximation in this case, we regularized the Hessian by adding to it the inverse covariance matrix of the flat prior, obtaining a matrix that would be the Hessian if the flat prior was replaced by a gaussian with the same mean and covariance. Although the gaussian with this regularized Hessian is still not a very good approximation for the posterior, we saw that in many cases, it improved the mixing rate of the chain.

In general, the multiplication of a vector of dimensionality d by a matrix involves $\mathcal{O}(d^2)$, and the inversion of a $d \times d$ matrix involves $\mathcal{O}(d^3)$ basic operations. In the decoding examples we consider, the dimension of $\mathbf{x}$ is most often proportional to the temporal duration of the stimulus. Thus, naively, the one-time inversion of J and calculation of A takes $\mathcal{O}(T^3)$ basic operations, where T is the duration of the stimulus, while the multiplication of $\mathbf{x}$ by A in each step of the MCMC algorithm takes $\mathcal{O}(T^2)$ operations. This would make the decoding of stimuli with even moderate duration forbidding. Fortunately, the quasi-locality of the GLM model allows us to overcome this limitation. Since the filters K_i in the GLM have a finite temporal duration, T_k, the Hessian of the GLM log-likelihood, equation 2.4, is banded in time: the matrix element $J^{\mathrm{LL}}_{t_1 n_1, t_2 n_2} \equiv -\partial^2 \log p(\boldsymbol{r} \mid \mathbf{x}) / \partial x(t_1, n_1) \partial x(t_2, n_2)$ vanishes when $\mid t_1 - t_2 \mid \geq 2T_k - 1$. The Hessian of the log posterior, equation 2.9, is the sum of the Hessians of the log prior and the log likelihood, which in the gaussian case is

$$J = J^{\mathrm{LL}} + C^{-1}, \tag{3.10}$$

where C is the prior covariance (see equation 2.10). Thus, if C^{-1} is also banded, J will be banded in time as well. As an example, gaussian autoregressive processes of any finite order form a large class of priors that have banded C^{-1}. In particular, for white noise stimuli, C^{-1} is diagonal, and therefore J will have the same bandwidth as J. Efficient algorithms can find the Cholesky decomposition of a banded $d \times d$ matrix, with bandwidth n_b, in a number of computations $\propto n_b^2 d$, instead of $\propto d^3$ (e.g., the command `chol` in Matlab uses the $\mathcal{O}(d)$ method automatically if J is banded and is encoded as a sparse matrix). Likewise, if B is a banded matrix with bandwidth n_b, the linear equation $B\mathbf{x} = \mathbf{y}$ can be solved for $\mathbf{x}$ in $\propto n_b d$ computations. Therefore, to calculate $\mathbf{x} = A\tilde{\mathbf{x}}$ from $\tilde{\mathbf{x}}$ in each step of the Markov chain, we proceed as follows. Before starting the chain, we first calculate the Cholesky decomposition of J such that $J = B^{\mathrm{T}} B$ and $\mathbf{x} = A\tilde{\mathbf{x}} = B^{-1}\tilde{\mathbf{x}}$. Then at each step of the MCMC, given $\tilde{\mathbf{x}}_t$, we find $\mathbf{x}_t$ by solving the equation $B\mathbf{x}_t = \tilde{\mathbf{x}}_t$. Since both of these procedures involve a number of computations that scale only with d (and thus with T), we can perform the whole MCMC decoding in $\mathcal{O}(T)$ computational time. This allows us to decode stimuli with durations on the order of many seconds. Similar methods with $\mathcal{O}(T)$ computational cost have been used previously in applications of MCMC to inference and estimation problems involving state-space models (Shephard & Pitt, 1997; Davis & Rodriguez-Yam, 2005; Jungbacker & Koopman, 2007), but these had not been generalized to non-state-space models (such as the GLM model we consider here) where the Hessian has a banded structure nevertheless. For a review of applications of state-space methods to neural data analysis see Paninski et al., 2010. That review also elucidates the close relationship between methods based on state-space models and methods exploiting the bandedness of the Hessian matrix, as described here. Exploiting the bandedness of the Hessian matrix in the optimization problem of finding the MAP is discussed in the companion article.

3.2 Hybrid Monte Carlo and MALA. A more powerful method for constructing rapidly mixing chains is the so-called hybrid or Hamiltonian Monte Carlo (HMC) method. In a sense, HMC is at the opposite end of the spectrum with respect to RWM in that it is designed to suppress the random walk nature of the chain by exploiting information about the local shape of $\pi(\mathbf{x})$, via its gradient, to encourage steps toward regions of higher probability. This method was originally inspired by the equations of Hamiltonian dynamics for the molecules in a gas (Duane, Kennedy, Pendleton, & Roweth, 1987) but has since been used extensively in Bayesian settings (for its use in sampling from posteriors based on GLM, see Ishwaran, 1999; see also Neal, 1996, for further applications and extensions).

This method starts with augmenting the vector $\mathbf{x}$ with an auxiliary vector of the same dimension $\mathbf{z}$. Let us define the potential energy as $\mathcal{E}(\mathbf{x}) = -\log \pi(\mathbf{x})$ up to a constant and a "Hamiltonian function" by $H(\mathbf{x}, \mathbf{z}) = \frac{1}{2}\mathbf{z}^{\mathrm{T}}\mathbf{z} + \mathcal{E}(\mathbf{x})$. Instead of sampling points, $\{\mathbf{x}_t\}$, from $\pi(\mathbf{x})$, the HMC method constructs an MH chain that samples points, $\{(\mathbf{x}_t, \mathbf{z}_t)\}$, from the joint distribution $p(\mathbf{x}, \mathbf{z}) \propto e^{-H(\mathbf{x},\mathbf{z})} \propto \exp(-\frac{1}{2}\mathbf{z}^{\mathrm{T}}\mathbf{z})\pi(\mathbf{x})$. But since this distribution is factorized into the products of its marginals for $\mathbf{x}$ and $\mathbf{z}$, the $\mathbf{x}$-part of the obtained samples yields samples from $\pi(\mathbf{x})$. On the other hand, sampling from the marginal over $\mathbf{z}$ is trivial, since $\mathbf{z}$ is normally distributed. In a generic step of the Markov chain, starting from $(\mathbf{x}_t, \mathbf{z}_t)$, the HMC algorithm performs the following steps to generate $(\mathbf{x}_{t+1}, \mathbf{z}_{t+1})$. First, to construct the MH proposal:

1) Set $\mathbf{x}_0 := \mathbf{x}_t$, and sample $\mathbf{z}_0$ from the isotropic gaussian distribution $\mathcal{N}_d(0, \mathbf{1})$.
2) Set $(\mathbf{x}, \mathbf{z}) := (\mathbf{x}_0, \mathbf{z}_0)$, and evolve $(\mathbf{x}, \mathbf{z})$ according to the equations of Hamiltonian dynamics[8] discretized based on the leapfrog method, by repeating the following steps L times
   - $\mathbf{z} := \mathbf{z} - \frac{\sigma}{2}\nabla\mathcal{E}(\mathbf{x})$
   - $\mathbf{x} := \mathbf{x} + \sigma\mathbf{z}$
   - $\mathbf{z} := \mathbf{z} - \frac{\sigma}{2}\nabla\mathcal{E}(\mathbf{x})$

---

[8] The continuous Hamiltonian equations are

$$\dot{\mathbf{z}} = -\frac{\partial H}{\partial \mathbf{x}} = -\nabla\mathcal{E}(\mathbf{x}), \qquad \dot{\mathbf{x}} = \frac{\partial H}{\partial \mathbf{z}} = \mathbf{z},$$

under which the Hamiltonian function is conserved.

Finally, to implement the MH acceptance step, equation 3.4,

3) With probability $\min\{1, \exp(-\Delta H)\}$, where $\Delta H \equiv H(\mathbf{x}, \mathbf{z}) - H(\mathbf{x}_0, \mathbf{z}_0)$, accept the proposal $\mathbf{x}$ as $\mathbf{x}_{t+1}$. Otherwise reject it, and set $\mathbf{x}_{t+1} = \mathbf{x}_t$. (It can be shown that this is a bona-fide MH rejection rule, ensuring that the resulting MCMC chain indeed has the desired equilibrium density; Duane et al., 1987.)

This chain has two parameters, L and σ, which can be chosen to maximize the mixing rate of the chain while minimizing the number of evaluations of $\mathcal{E}(\mathbf{x})$ and its gradient. In practice, even a small L, requiring fewer gradient evaluations, often yields a rapidly mixing chain, and therefore in our simulations, we used $L \in \{1, \ldots, 5\}$. The special case of $L = 1$ corresponds to a chain that has proposals of the form

$$\mathbf{y} \sim \mathbf{x} - \frac{\sigma^2}{2}\nabla\mathcal{E}(\mathbf{x}) + \sigma\mathbf{z}, \tag{3.11}$$

where $\mathbf{z}$ is normal with zero mean and identity covariance, and the proposal $\mathbf{y}$ is accepted according to the MH rule, equation 3.4. In the limit $\sigma \to 0$, this chain becomes a continuous Langevin process with the potential function $\mathcal{E}(\mathbf{x}) = -\log\pi(\mathbf{x})$, whose stationary distribution is the Gibbs measure, $\pi(\mathbf{x}) = \exp(-\mathcal{E}(\mathbf{x}))$, without the MH rejection step. For a finite σ, however, the Metropolis-Hastings acceptance step is necessary to guarantee that $\pi(\mathbf{x})$ is the invariant distribution. The chain is thus referred to as the Metropolis-adjusted Langevin algorithm (MALA) (Roberts & Tweedie, 1996).

The scale parameter σ, which also needs to be adjusted for the RWM chain, sets the average size of the proposal jumps: we must typically choose this scale to be small enough to avoid jumping wildly into a region of low $\pi(\mathbf{x})$, and therefore wasting the proposal, since it will be rejected with high probability. At the same time, we want to make the jumps as large as possible, on average, in order to improve the mixing time of the algorithm. (See Roberts and Rosenthal, 2001, and Gelman, 2004, for some tips on how to find a good balance between these two competing desiderata for the RWM and MALA chains.) For the HMC chains with $L > 1$, we chose σ, by trial and error, to obtain an MH acceptance rate of 60% to 70%. We adopted this rule of thumb, based on a qualitative extrapolation of the results of Roberts and Rosenthal (1998), for the special cases of $L = 0$ and 1 (corresponding to the RWM and MALA chains, respectively), and their suggestion to tune the acceptance rate in those cases to ~25% and ~55%, respectively, for optimal mixing (for further discussion, see section 3.5; for a study on tuning the σ parameter for HMC with general L, see, e.g., Kennedy, Edwards, Mino, & Pendleton, 1996).

For highly nonisotropic distributions, the HMC chains can also be enhanced by exploiting the Laplace approximation (or its regularized version in the uniform prior case, as explained in the RWM case) by modifying the HMC proposals. Equivalently, as noted after equation 3.9, we can sample from the auxiliary distribution $\tilde{\pi}(\tilde{\mathbf{x}}) = \mid A \mid \pi(A\tilde{\mathbf{x}})$ (where A is given in equation 3.9) using the unmodified HMC chain, described above, and subsequently transforming the samples by A. As explained in the final paragraph of section 3.1, we can perform this transformation efficiently in $\mathcal{O}(T)$ computational time, where T is the stimulus duration. Another practical advantage of this transformation by A is that the process of finding the appropriate scale parameter σ simplifies considerably, since $\tilde{\pi}(\tilde{\mathbf{x}})$ may be approximated as a gaussian distribution with identity covariance regardless of the scaling of different dimensions in the original distribution $\pi(\mathbf{x})$. To our knowledge, this $\mathcal{O}(T)$ enhancement of the HMC chain using the Laplace approximation is novel. This chain turned out to be the most efficient in most of the decoding examples we explored (we discuss this in more detail in section 3.5).

It is worth noting that when sampling from high-dimensional distributions with sharp gradients, the MALA, HMC, and RWM chains have a tendency to be trapped in "corners" where the log posterior changes suddenly. This is because when the chain eventually ventures close to the corner, a jump proposal will very likely fall on the exterior side of the sharp high-dimensional corner (the probability of jumping to the interior side from the tip of a cone decreases exponentially with increasing dimensionality). Thus, most proposals will be rejected, and the chain will effectively stop. As we will see below, the hit-and-run chain is known to have an advantage in escaping from such sharp corners (Lovasz & Vempala, 2004). We discuss this point further in section 3.4.

3.3 The Gibbs Sampler. Gibbs sampling (Geman & Geman, 1984) is an important MCMC scheme. It is particularly efficient when, despite the complexity of the distribution $\pi(\mathbf{x}) = p(\mathbf{x} \mid \boldsymbol{r}, \theta)$, its one-dimensional conditionals $p(x_m \mid \mathbf{x}_{\perp m}, \boldsymbol{r}, \theta)$ are easy to sample from. Here, x_m is the mth component of $\mathbf{x}$, and $\mathbf{x}_{\perp m}$ denotes the other components, that is, the projection of $\mathbf{x}$ on the subspace orthogonal to the mth axis. The Gibbs update is defined as follows. First, choose the dimension m randomly or in order. Then update $\mathbf{x}$ along this dimension, that is, sample x_m from $\pi(x_m \mid \mathbf{x}_{\perp m})$ (while leaving the other components fixed). This is equivalent to sampling a one-dimensional auxiliary variable, s, from

$$s \sim h(s \mid m, \mathbf{x}) \propto \pi(\mathbf{x} + s\mathbf{e}_m), \qquad -\infty < s < \infty, \tag{3.12}$$

and setting $\mathbf{y} = \mathbf{x} + s\mathbf{e}_m$, where $\mathbf{e}_m$ is the unit vector along the mth axis (we discuss how to sample from this one-dimensional distribution in section 3.4). It is well known that the Gibbs rule is indeed a special case of the MH algorithm where the proposal, equation 3.12, is always

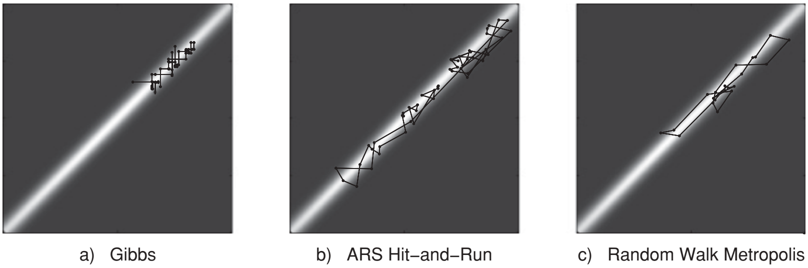

Figure 3: Comparison of different MCMC algorithms in sampling from a nonisotropic truncated gaussian distribution. This distribution can arise as a posterior distribution resulting from a nonisotropic gaussian likelihood and a uniform prior with square boundaries (at the frame borders). (a–c) Fifty-sample chains for a Gibbs, isotropic hit-and-run and isotropic random walk Metropolis (RWM) samplers, respectively. The grayscale indicates the height of the probability density. As seen in a, the narrow nonisotropic likelihood can significantly hamper the mixing of the Gibbs chain as it chooses its jump directions unfavorably. The hit-and-run chain mixes much faster as it samples the direction randomly and hence can move within the narrow high-likelihood region with relative ease. The mixing of the RWM chain is relatively slower due to its rejections (note that there are fewer than 50 distinct dots in c due to rejections; the acceptance rate was about 0.4 here). For illustrative purposes, the hit-and-run direction and the RWM proposal distributions were taken to be isotropic here, which is disadvantageous, as explained in the text (also see Figure 2).

accepted. (For applications of the Gibbs algorithm for sampling from posterior distributions involving GLM-like likelihoods, see Chan & Ledolter, 1995, and Gamerman, 1997, 1998; see also Smith, Wirth, Suzuki, & Brown, 2007, for some related applications in neural data analysis, discussed in section 5.1.)

It is important to note that the Gibbs update rule can sometimes fail to lead to an ergodic chain; the chain can get "stuck" and not sample from $\pi(\mathbf{x})$ properly (Robert & Casella, 2005). An extreme case of this is when the conditional distributions $p_m(x_m \mid \mathbf{x}_{\perp m}, \boldsymbol{r}, \theta)$ are deterministic: then the Gibbs algorithm will never move, clearly breaking the ergodicity of the chain. More generally, in cases where strong correlations between the components of $\mathbf{x}$ lead to nearly deterministic conditionals, the mixing rate of the Gibbs method can be extremely low (Figure 3a shows this phenomenon for a two-dimensional distribution with strong correlation between the two components). Thus, it is a good idea to choose the parameterization of the model carefully before blindly applying the Gibbs algorithm. For example, we can change the basis or, more systematically, exploit the Laplace approximation, as described above, to sample from the auxiliary distribution $\tilde{\pi}(\tilde{\mathbf{x}})$ instead.

3.4 The Hit-and-Run Algorithm. The hit-and-run algorithm (Boneh & Golan, 1979; Smith, 1980; Lovasz & Vempala, 2004) can be thought of as random-direction Gibbs: in each step of the hit-and-run algorithm, instead of updating $\mathbf{x}$ along one of the coordinate axes, we update it along a random general direction not necessarily parallel to any coordinate axis. More precisely, the sampler is defined in two steps. First, choose a direction $\mathbf{n}$ from some positive density $\rho(\mathbf{n})$ (with respect to the normalized Lebesgue measure) on the unit sphere $\mathbf{n}^{\mathrm{T}}\mathbf{n} = 1$. Then, similar to Gibbs, sample the new point on the line defined by $\mathbf{n}$ and $\mathbf{x}$, with a density proportional to the underlying distribution. That is, sample s from

$$s \sim h(s \mid \mathbf{n}, \mathbf{x}) \propto \pi(\mathbf{x} + s\mathbf{n}), \qquad -\infty < s < \infty, \tag{3.13}$$

and set $\mathbf{y} = \mathbf{x} + s\mathbf{n}$.[9] Although the hit-and-run chain is well known in the statistics literature, it has not been used in neural decoding.

The main gain over RWM or HMC is that instead of taking small local steps (of size proportional to σ, in equation 3.7 or 3.11), we may take very large jumps in the $\mathbf{n}$ direction; the jump size is set by the underlying distribution itself, not an arbitrary scale, σ, which has to be tuned by the user to achieve optimal efficiency. The jump size in a given direction, $\mathbf{n}$, is set by the scale of the "slice" distribution, equation 3.13.

This, together with the fact that all hit-and-run proposals are accepted, makes the chain better at escaping from sharp high-dimensional corners (see Lovasz & Vempala, 2004, and the discussion at the end of section 3.2 above). The advantage over Gibbs is in situations such as depicted in Figure 2, where jumps parallel to coordinates lead to small steps, but there are directions that allow long jumps to be made by hit-and-run. The price to pay for these possibly long nonlocal jumps, however, is that now (as well as in the Gibbs case) we need to sample from the one-dimensional density $\frac{1}{Z}\pi(\mathbf{x} + s\mathbf{n})$, which is in general nontrivial. Fortunately, as we mentioned above (see the discussion leading to equations 2.5 to 2.7 and following it), in the case of neurons modeled by the GLM, the posterior distribution and thus all its "slices" are log concave, and efficient methods such as adaptive rejection sampling (ARS) (Gilks, 1992; Gilks & Wild, 1992) can be used to sample from the one-dimensional slice in the hit-and-run step. Let us emphasize, however, that the hit-and-run algorithm by itself does not require the distribution $\pi(\mathbf{x})$ to be log concave. Given a method other than ARS for sampling from the one-dimensional conditional distributions, $\pi(\mathbf{x} + s\mathbf{n})$, hit-and-run can be applied to general distributions that are not log concave as well.

---

[9] As with the Gibbs case, it can be shown again that this proposal leads to an MH acceptance probability of one. Hence, hit-and-run is also a special case of MH.

Regarding the direction density, $\rho(\mathbf{n})$, the easiest choice is the isotropic $\rho(\mathbf{n}) = 1$. More generally it is easy to sample from ellipses by sampling from the appropriate gaussian distribution and normalizing. Thus, again, a reasonable approach is to exploit the Laplace approximation: we sample $\mathbf{n}$ by sampling an auxiliary point $\tilde{\mathbf{x}}$ from $\mathcal{N}(0, J^{-1})$, where J is the Hessian, equation 2.9, and setting $\mathbf{n} = \tilde{\mathbf{x}}/\|\tilde{\mathbf{x}}\|$ (see Figure 2). This prescription is equivalent to sampling $\mathbf{n}$ from the distribution $\rho(\mathbf{n}) = \sqrt{\det J/(\mathbf{n}^{\mathrm{T}} J\, \mathbf{n})^{d}}$, which is referred to as the angular central gaussian distribution in the statistical literature (see e.g., Tyler, 1987). This further advantages hit-and-run over Gibbs by giving more weight to directions that allow larger jumps to be made.

3.5 Comparison of Different MCMC Chains. Above, we pointed out some qualitative reasons behind the strengths and weaknesses of the different MCMC algorithms in terms of their mixing rates and computational costs. Here we give a more quantitative account and also compare the different methods based on their performance in the neural decoding setting.

From a practical point of view, the most relevant notion of mixing is how fast the estimate $\hat{g}_N^{(\pi)}$ of equation 3.2 converges to the true expectation of the quantity of interest, f. As one always has access to finitely many samples, N, even in the optimal case of i.i.d. samples from π, $\hat{g}_N^{(\pi)}$ has a finite random error, equation 3.3. For the correlated samples of the MCMC chain and for large N, the error is larger, and equation 3.3 generalizes to (see Kipnis & Varadhan, 1986)

$$\mathrm{Var}(\hat{g}_N) = \frac{\tau_{\mathrm{corr}}}{N}\mathrm{Var}[g(\mathbf{x})] + o\left(\frac{\tau_{\mathrm{corr}}}{N}\right), \tag{3.14}$$

for $N \gg \tau_{\mathrm{corr}}$, independent of the starting point.[10] Here, τ_{corr} is the equilibrium autocorrelation time of the scalar process $g(\mathbf{x}_i)$, based on the chain $\mathbf{x}_i$. It is defined by

$$\tau_{\mathrm{corr}} = \sum_{t=-\infty}^{\infty} \gamma_t \equiv \sum_{t=-\infty}^{\infty} \mathrm{Corr}\left(g(\mathbf{x}_i)g(\mathbf{x}_{i+t})\right), \tag{3.15}$$

where we refer to γ_t as the lag-t autocorrelation for $g(\mathbf{x})$. Thus, the smaller the τ_{corr}, the more efficient is the MCMC algorithm, as one can run a shorter chain to achieve a desired estimated error.

---

[10] Strictly speaking this independence is true only for Harris recurrent chains, but this is the case in most practical examples (see e.g., Geyer, 1992, and Tierney, 1991).

Another measure of mixing speed, which has the merit of being more amenable to analytical treatment, is the mean squared jump size of the Markov chain,

$$a^2 = E(\|\mathbf{x}_{t+1} - \mathbf{x}_t\|^2), \tag{3.16}$$

termed the first-order efficiency (FOE) by Roberts and Rosenthal (1998). Let us define $\gamma^m_{t=1}$ to be the lag-1 autocorrelation of the mth component of $\mathbf{x}$, x_m. From equation 3.16, it follows that the weighted average of $\gamma^m_{t=1}$ over all components (with weights $\mathrm{Var}[x_m]$), is given by $1 - \frac{a^2}{2\sum_m \mathrm{Var}[x_m]}$. Thus, maximizing the FOE is roughly equivalent to minimizing correlations. One analytical result concerning the mixing performance of different MCMC chains was obtained in Roberts and Rosenthal (1998) for the FOE of RWM and MALA when sampling from the restricted class of product distributions $\pi(\mathbf{x}) = \prod_{m=1}^d g(x_m)$, and asymptotically large dimension $d = \dim(\mathbf{x})$ (often a relevant limit in neural decoding). Based on their results, the authors also argue that in general, the jump scales of RWM and MALA proposals may be chosen such that their acceptance rates are roughly 0.25 and 0.55, respectively. For the special case of sampling from a d-dimensional standard gaussian distribution, $\pi(\mathbf{x}) \propto \exp(-\|\mathbf{x}\|^2/2)$ and for optimally chosen proposal jump scales, they show that the FOE of gaussian MALA and RWM are asymptotically equal to $1.6d^{2/3}$ and 1.33, respectively.

To enable a comparison with hit-and-run, we can calculate its FOE directly. Using $\mathbf{y} = \mathbf{x} + s\mathbf{n}$, with s sampled as in equation 3.13, we see that

$$a^2 = \int\int E\left(s^2 \mid \mathbf{n}, \mathbf{x}\right) \rho(\mathbf{n})\pi(\mathbf{x})\, d\mathbf{n}\, d\mathbf{x}. \tag{3.17}$$

Now, from equation 3.13, $h(s \mid \mathbf{n}, \mathbf{x}) \propto e^{-\frac{(\|\mathbf{x}\|^2 + s^2 + 2s\mathbf{n}\cdot\mathbf{x})^2}{2}} \propto e^{-\frac{(s-\mathbf{n}\cdot\mathbf{x})^2}{2}}$, and using $E\left(s^2 \mid \mathbf{n}, \mathbf{x}\right) = E\left(s \mid \mathbf{n}, \mathbf{x}\right)^2 + \mathrm{Var}\left(s \mid \mathbf{n}, \mathbf{x}\right)$, we obtain $E\left(s^2 \mid \mathbf{n}, \mathbf{x}\right) = (\mathbf{n}\cdot\mathbf{x})^2 + 1$. Thus,

$$a^2 = \int E_\pi\left((\mathbf{n}\cdot\mathbf{x})^2 + 1\right) \rho(\mathbf{n})\, d\mathbf{n}, \tag{3.18}$$

$$= \int (\mathbf{n}\cdot\mathbf{n} + 1)\, \rho(\mathbf{n})\, d\mathbf{n} = 2, \tag{3.19}$$

where we used $E_\pi(x_n x_m) = \delta_{nm}$ for the standard gaussian distribution $\mathcal{N}_d(0, \mathbf{1})$ and $\mathbf{n}\cdot\mathbf{n} = 1$. Therefore, while hit-and-run has higher FOE than RWM in this case, we see that for unimodal, nearly gaussian distributions, MALA will mix much faster (by a factor $\propto d^{2/3}$) than both RWM and hit-and-run in large dimensions. Although we know of no such result for general HMC chains with higher-order leapfrog steps than the one-step

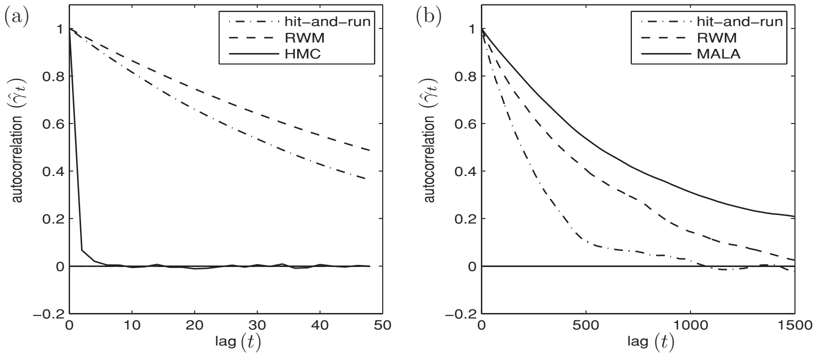

Figure 4: The estimated autocorrelation function for the hit-and-run, gaussian random walk metropolis, and HMC chains, based on seven separate chains in each case. The chains were sampling from a posterior distribution over a 50-dimensional stimulus (**x**) space with white noise gaussian (a) and uniform (b) priors with contrast $c = 1$ (see equations 2.10 and 2.11), and with GLM likelihood (see equations 2.3 and 2.4) based on the response of two simulated ganglion cells. The GLM nonlinearity was exponential, and the stimulus filters $k_i(t)$ were taken to be weak delta functions with heights ±0.1. For the HMC, we used $L = 5$ leapfrog steps in the gaussian prior case and $L = 1$ steps (corresponding to MALA) in the flat prior case. The autocorrelation was calculated for a certain one-dimensional projection of **x**. In general, in the gaussian prior case, HMC was superior by an order of magnitude. For uniform priors, however, hit-and-run was seen to mix faster than the other two chains over a wide range of parameters such as the stimulus filter strength (unless the filter was strong enough so that the likelihood determined the shape of the posterior, confining its effective support away from the edges of the flat prior). This is mainly because hit-and-run is better in escaping from the sharp, high-dimensional corners of the prior support $\mathcal{S}$. Here, MALA need not be slower than RWM, and its larger autocorrelation in the plot is because its jump size was chosen suboptimally, according to a rule (Roberts & Rosenthal, 1998) that is optimal only for smooth distributions. For both priors, using nonisotropic proposal or direction distributions improved the mixing of all three chains.

MALA algorithm, we expect their mixing speed to increase even further for higher leapfrog steps. The superiority of HMC over the other chains is clearly visible in Figure 4a, which shows a plot of the estimated autocorrelation function γ_t for the sampling of the three chains from the GLM posterior with standard gaussian priors, and a weak stimulus filter leading to a weak likelihood. More generally, in our simulations with gaussian priors and smooth GLM nonlinearities, HMC (including MALA) had an order-of-magnitude advantage over the other chains for most of the relevant parameter ranges. Thus, we used this chain in section 5.2 for evaluating the mutual information with gaussian priors.

The situation can be very different, however, for highly nongaussian (but still log-concave) distributions, such as those with sharp boundaries. In our GLM setting, this can be the case with flat priors on convex sets, equation 2.11, when the likelihood is broad and does not restrict the posterior support away from the boundaries and corners of the prior support $\mathcal{S}$. In this case, HMC and MALA lose their advantage because they do not take advantage of the information in the prior distribution, which has zero gradient within its support. Furthermore, as mentioned in sections 3.2 and 3.4, when the convex body $\mathcal{S}$ has sharp corners, hit-and-run will have an advantage over both RWM and HMC in avoiding getting trapped in those corners, which can otherwise considerably slow the chain in large dimensionality (see the arguments in Lovasz & Vempala, 2004). Finally, the MALA or HMC proposals can in principle be inefficient in regions of sharp gradient changes; for example, in the GLM setting, if the nonlinearity $f(.)$ is very sharp, then the log likelihood might vary much more quickly than quadratic. In such cases, the HMC proposal jumps can be too large, falling in regions where $\pi(\mathbf{x})$ is very low and leading to high rejection rates. This can potentially reduce HMC's advantage significantly even in case that the prior is gaussian. However, in our experience, with $f(.) = \exp(.)$, this did not occur.

Figure 4b shows the estimated autocorrelation function for different chains in sampling from the posterior distribution in GLM-based decoding with a flat prior stimulus distribution, equation 2.11, with cubic support.[11] For this prior, the correlation time of the hit-and-run chain was consistently lower than those of the RWM, MALA, and Gibbs (not shown in the figure) chains, unless the likelihood was sharp and concentrated away from the boundaries of the prior cube. As we mentioned above (also see the next section), the Laplace approximation is adequate in this latter case. Thus, we see that hit-and-run is the faster chain when this approximation fails, which is also the case where MCMC is more indispensable. We thus used the hit-and-run algorithm in our decoding examples for the flat prior case presented in the next section.

Finally, we note that other methods of diagnosing mixing and convergence, such as the so-called r-hat ($\hat{R}$) statistic (Brooks & Gelman, 1998), gave consistent results with those based on the autocorrelation time, τ_{corr}, presented here.

4 Comparison of MAP and Monte Carlo Decoding

In this section we compare Bayesian stimulus decoding using the MAP and the posterior mean estimates, equations 2.7 and 3.1, based on the response of a population of neurons modeled via the GLM introduced in section 2. We will show that in the flat prior case, equation 2.11, the MAP estimate, in terms of its mean squared error, is much less efficient than the posterior mean estimate. We contrast this with the gaussian prior case, where the Laplace approximation is accurate in a large range of model parameters, and thus the two estimates are close. Furthermore, for both kinds of priors, in the limit of strong likelihoods (e.g., due to a strong stimulus filter or a large number of neurons), the posterior distribution will be sharply concentrated, the Laplace approximation becomes asymptotically more and more accurate, and both estimates will eventually converge to the true stimulus (more precisely, the part of the stimulus that is not outside the receptive field of all the neurons; see footnote 13).

---

[11] Although this prior belongs to the class of product densities considered in Roberts and Rosenthal (1998), it does not satisfy the stringent smoothness conditions crucial for the part of their theorem regarding the (fast) mixing of MALA.

In the first two examples (Figures 5 and 6), the stimulus estimates were computed given the simulated spike trains of a population of pairs of ON and OFF retinal ganglion cells (RGC) in response to a spatially uniform, full-field fluctuating light intensity signal. The stimuli were discretized white noise with gaussian and flat distributions (see the paragraph after equation 2.11). Spike responses were generated by simulating the GLM point process encoding model, described by equations 2.3 and 2.4, with exponential nonlinearity, $f(u) = \exp(u)$. The coupling between different cells ($\mathcal{H}_{ij}$ of equation 2.3 for $i \neq j$) was set equal to zero, but the diagonal kernels, $\mathcal{H}_{ii}$, representing the spike history feedback of each cell to itself were closely matched to those found with fits to macaque ON and OFF RGC's reported in Pillow et al. (2008), and so were the DC biases, b_i; the values of the DC biases were such that the baseline firing rate, $\exp(b_i)$, in the absence of stimulus was approximately 7 Hz (see the appendix of Pillow et al., 2011, the companion article, for a more detailed description of the fits for stimulus and spike history filters). However, for demonstration purposes, the stimulus filters, K_i, were set to positive and negative delta functions (for ON and OFF cells, respectively), resulting in $K_i \cdot \mathbf{x}$ being proportional to the light stimulus, $x(t)$, so that bandpass filtering of the stimulus did not result in information loss and convergence of the estimates to the true stimulus could be observed more easily. For a fixed number of cells, the parameter of relevance here, which determines the signal-to-noise ratio of the RGCs' spike trains, is the strength of the filtered stimulus input, $K_i \cdot \mathbf{x}$, to the GLM nonlinearity. The magnitude of this input is proportional to $c\|k\|$, where c is the stimulus contrast and $\|k\|$ is the norm of the receptive field filter (which we have taken to be the same for all cells in this example). Figure 5 shows the stimulus, the spike trains, and the two estimates for three different magnitudes of $c\|k\|$, based on the response of one pair of ON and OFF cells. Figure 6 shows the same based on the response of 10 identical pairs of RGCs.

Because the prior distribution here is flat on the 50-dimensional cube centered at the origin, the Laplace approximation, equation 2.8, will be justified only when the likelihood is sharply concentrated and supported away from

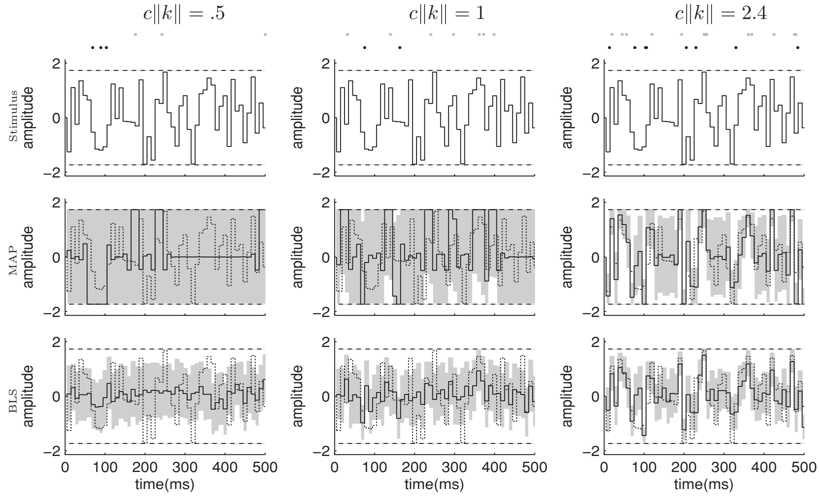

Figure 5: Comparison of MAP and posterior mean estimates, for a pair of ON and OFF RGCs (see the main text), for different values of the stimulus filter amplitude ($\|k\| = 0.5$, 1, and 2.4 from left to right) and contrast $c = 1$ (defined after equation 2.11. The product $c\,\|k\|$ represents the scale of the filtered stimulus input term to the GLM nonlinearity (see the main text for the full description of the GLM parameters used in this simulation). The stimulus (black traces in the first row panels and dotted traces in other rows) consists of a 500 ms interval of uniformly distributed white noise, refreshed every 10 ms. Thus, the stimulus space is 50-dimensional. The dashed horizontal lines mark the boundaries of the flat prior distribution of the stimulus intensity on each 10 ms subinterval. They are set at $\pm\sqrt{3}$, corresponding to intensity variance of 1 and zero mean. Dots on the top row show the spikes of the ON (gray) and the OFF (black) cell. The solid traces in the middle row are the MAP estimates, and the solid traces in the bottom row show the posterior means estimated from 10,000 samples of a hit-and-run chain (after burning 2500 samples). The shaded regions in the second and third rows are error bars showing the estimated marginal posterior uncertainties about the stimulus value. For the MAP (second rows), these are calculated as the square root of the diagonal of the inverse Hessian, J^{-1}, but they have been cut-off where they would have encroached on the zero prior region beyond the horizontal dashed lines. For the posterior mean (third rows), the error bars represent one standard deviation about the mean and are calculated as the square root of the diagonal of the covariance matrix, which is itself estimated from the MCMC chain (the standard error of the posterior mean estimate due to the finite sample size of the MCMC were much smaller than these error bars, and are not shown). Note that the error bars of the mean are in general smaller than those for the MAP and that all posterior uncertainties decrease as the stimulus filter amplitude grows.

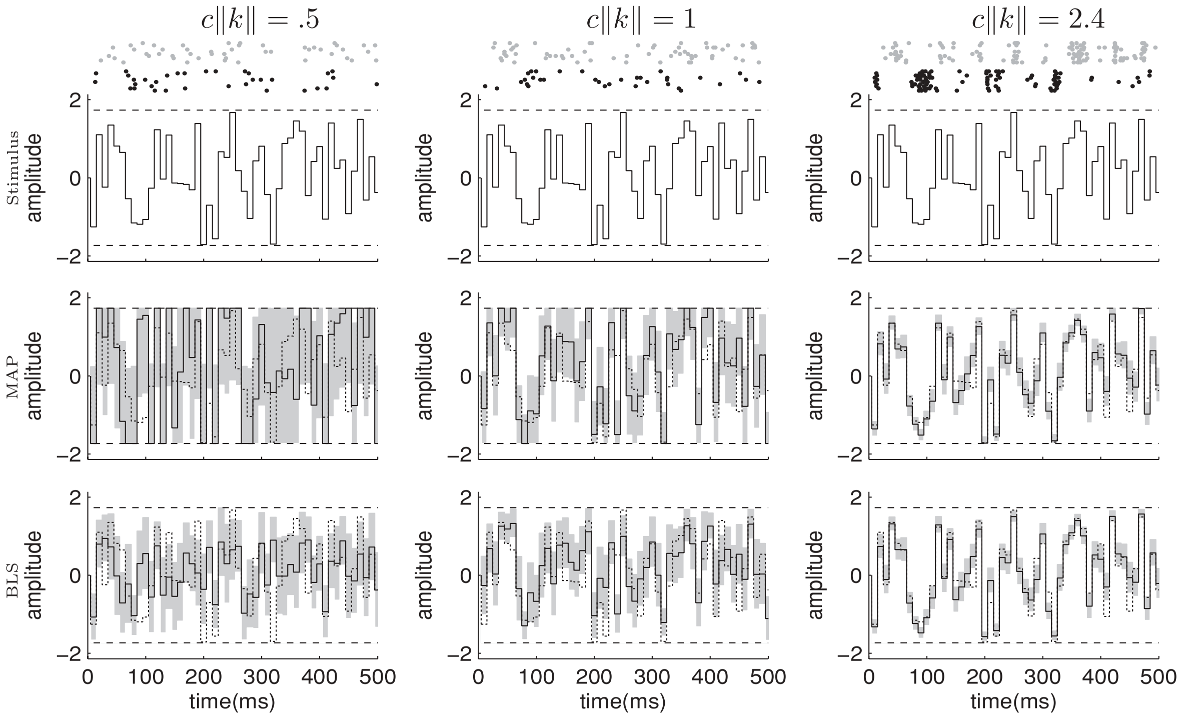

Figure 6: Comparison of MAP and posterior mean estimates for 10 identical and independent pairs of ON and OFF RGCs for different values of the stimulus filter. The stimulus and all GLM parameters are the same as in Figure 5, except for the number of pairs of RGC. The increase in the number of cells leads to the sharpening of the likelihood, leading to smaller posterior uncertainties for the decoded estimates, and a more accurate Laplace approximate and smaller disparity between the two decoders. Here a 20,000-sample-long MALA chain (after burning 5000 samples) was used to estimate the posterior mean.

the edges of the cube.[12] Moreover, since the flat prior is only "felt" on the boundaries of the cube (the horizontal dashed lines in Figures 5 and 6), the MAP will lie in the interior of the cube only if the likelihood has a maximum there. For filtered stimulus inputs with small magnitude, $c\|k\|$, the log likelihood, equations 2.3 and 2.4, becomes approximately linear in the components of $\mathbf{x}$. With a flat prior, the almost linear log posterior will very likely be maximized only on the boundaries of the cube (since linear functions on convex domains attain their maxima at the "corners" of the domain). Thus, in the absence of a strong, confining likelihood, the MAP has a tendency to stick to the boundaries, as seen in the first two columns of Figure 5; in other words, the MAP falls on a corner of the cube, where the Laplace approximation is worst and where MALA and RWM are least efficient. We note that the likelihood will be further weakened, in fact, if we replace the delta function stimulus filters with more realistic filters, as the bandpass filtering will remove the dependence of the likelihood on the features of the stimulus that were filtered out (see a similar discussion in our companion article, Pillow et al., 2011, on MAP decoding).

A sharp likelihood confines the posterior away from the boundaries of the prior support and solely determines the position of both the MAP and the posterior mean. In this case, the gaussian approximation for the posterior distribution is valid, and the two estimates will in fact be very close (as the mean and the mode of a gaussian are one and the same). This can be seen in the right column of Figure 5, where the large value of the stimulus filter has sharpened the likelihood. Also, as is generally true in statistical parameter estimation, when the number of data points becomes large, the likelihood term gets very sharp, leading to accurate estimates.[13] In our case, this corresponds to increasing the number of cells with similar receptive fields, leading to the smaller error bars in Figure 6 and the more accurate and closer MAP and mean estimates.

To compare the performance of the two estimates more quantitatively, in Figure 7, we have plotted the average squared errors of the two estimates under the full stimulus-response distribution, $p(\mathbf{x}, \boldsymbol{r})$ (for the same type of stimulus and cell pair as in the Figure 5 simulations), as a function of the magnitude of the filtered stimulus input, $c\|k\|$. This was done by generating five samples of the stimulus in each case and then simulating the GLM to

---

[12] More precisely, "sharply concentrated" here means that the curvature of the log posterior is large enough so that the Taylor expansion of the log posterior involved in the Laplace approximation, equation 2.8, is accurate for deviations from $\mathbf{x}_{\mathrm{MAP}}$ on a scale determined by the inverse square root of the smallest eigenvalue of the Hessian matrix, equation 2.9.

[13] This is obviously not the case, however, for parameter directions along which the data are noninformative and the likelihood function does not vary much. In the RGC case, these correspond to stimulus features (directions in the stimulus space) that fall orthogonal to the cells' spatiotemporal filters K_i, and are hence filtered out.

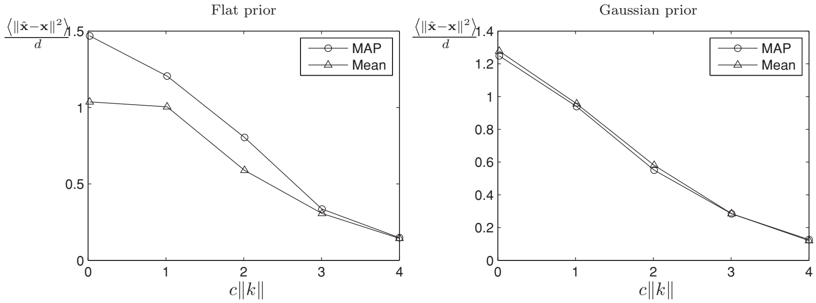

Figure 7: Comparison of mean squared error ($\langle \|\hat{\mathbf{x}} - \mathbf{x}\|^2 \rangle / d$) of MAP and posterior mean estimates for uniform (left panel) and gaussian (right panel) white noise stimulus distributions as a function of the stimulus filter strength times contrast. In the left panel, the data points at $\|k\| = 0$ were obtained for very small but nonzero $\|k\|$. As seen here, for flat priors, MAP has a higher average squared error than the posterior mean, except for large values of the stimulus filter where both estimates converge to the true value. For gaussian priors, the Laplace approximation is accurate, and therefore the posterior mean and MAP are very close. Thus, their efficiency (e.g., as measured by the inverse of their mean squared error) is very similar even for small values of the stimulus filter, and the fact that the computational cost of calculating MAP is much lower makes it the preferable estimate here.

generate the spike train response of the pair of ON and OFF cells to each stimulus, leading to sample pairs $(\mathbf{x}_i, \boldsymbol{r}_i)$ for $i = 1, \ldots, 5$. For each of the responses, $\boldsymbol{r}_i$, the MAP and MCMC mean were computed based on the posteriors $p(\mathbf{x} \mid \boldsymbol{r}_i)$. The average (over $p(\mathbf{x}, \boldsymbol{r})$) square error, $\langle \|\hat{\mathbf{x}}(\boldsymbol{r}) - \mathbf{x}\|^2 \rangle$, was then approximated by its sample mean, $\sum_{i=1}^{5} \|\hat{\mathbf{x}}(\boldsymbol{r}_i) - \mathbf{x}_i\|^2/5$. The left and right panels in Figure 7 show plots of the squared error per dimension, for MAP and mean estimates, as a function of the stimulus filter strength for the case of the flat and gassian white noise stimulus ensembles, respectively. As is evident from the plots, in the former ensemble, the MAP is inferior to the mean, due to its higher mean squared error, unless the filter strength is large. For the gaussian ensemble, the plot shows that the error of the MAP and posterior mean estimates are very close throughout the range of stimulus filter strength. Thus, due to its much lower computational cost, the MAP-based decoding method of Pillow et al. (2011) in the companion article is superior for this prior. Let us mention that the magnitude of the filtered stimulus, $c\|k\|$, in the experimental data reported in Pillow et al. (2008) (which is also the basis of the final example in this section—see Figure 8) was in the range 3 ± 1, depending on the cell; smaller values of $c\|k\|$ can be achieved experimentally by lowering the contrast of the visual stimulus as needed. Thus, the values of this parameter used in Figure 7, as well as in Figures 5 and 6, are on the same order of magnitude as those used in that experiment and cover a range of values that is experimentally and biologically relevant.

Finally, we compared the MAP and posterior mean estimates in decoding the experimentally recorded spike trains. The spike trains were recorded from a group of 11 ON and 16 OFF RGCs (whose receptive fields fully cover a patch of the visual field) in response to the light signal of the optically reduced image of a cathode ray display that refreshes at 120 Hz and is projected on the retina (Litke et al., 2004; Shlens et al., 2006). The stimlulus, $\mathbf{x}$, in this case, is a spatiotemporally fluctuating binary white noise, with $x(t, n)$ representing the contrast of the pixel n at time t. In Pillow et al. (2008), 20 minutes of these data were used to fit the GLM model parameters including cross-couplings, h_{ij}, to these cells (see that reference for details about the recording and the fitting method and a full description of the fit GLM parameters). Here, we took a 500 ms portion of the recorded spike trains of six neighboring RGCs (three ON and three OFF) and, using the fit GLM parameters for them, decoded the filtered inputs,

$$\mathbf{y}_i \equiv K_i \cdot \mathbf{x}, \tag{4.1}$$

to these cells using the MAP and posterior mean (calculated using an HMC chain). The inputs are a priori correlated due to the overlaps between the cell's receptive fields, and the covariance matrix of the $\mathbf{y}_i$ is given by $\mathcal{C}_y^{ij} = K_i \mathcal{C}_x K_j^{\mathrm{T}}$, where $\mathcal{C}_x = c^2 \mathbf{1}$ is the covariance of the white noise visual stimulus. More explicitly,

$$\mathcal{C}_y(i, t_1; j, t_2) \equiv \mathrm{Cov}[y_i(t_1), y_j(t_2)] = c^2 \sum_{t,n} k_i(t_1 - t, n) k_j(t_2 - t, n). \tag{4.2}$$

Notice that with the experimentally fit k_i, which have a finite temporal duration T_k, the covariance matrix, $\mathcal{C}_y$ is banded: it vanishes when $|t_1 - t_2| \geq 2T_k - 1$. Since $\mathbf{x}$ is binary, $\mathbf{y}_i$ is not a gaussian vector. However, because the filters $K_i(t, n)$ have a relatively large spatiotemporal dimension, $\mathbf{y}_i(t)$ are weighted sums of many i.i.d. binary random variables, and their prior marginal distributions can be well approximated by gaussian distributions. For this reason and because the likelihood was relatively strong for these data (and hence the dependence on the prior relatively weak), we replaced the true (highly nongaussian) joint prior distribution of $\mathbf{y}_i$ with a gaussian distribution with zero mean and covariance, equation 4.2. This allowed us to implement the efficient nonisotropic HMC chain, described above, so that its computational cost scales only linearly with the stimulus duration T, allowing us to decode very long stimuli. However, in this case, the details of the procedure explained in the final paragraph of section 3.1 have to be modified as follows. The Hessian for $\mathbf{y}$ is given by

$$J_y = C_y^{-1} + J_y^{\mathrm{LL}}, \tag{4.3}$$

where the Hessian of the negative log likelihood term, J_y^{LL}, is now diagonal, because $y_i(t)$ affects the conditional firing rate instantaneously (see equation 2.3). Let $AA^{\mathrm{T}} = J_y^{-1}$, similar to equation 3.9. The nonisotropic chain requires the calculation of $A\tilde{\mathbf{y}}$ for some vector $\tilde{\mathbf{y}}$ at each step of the MCMC. In order to carry this out in $\mathcal{O}(T)$ computational time, we proceed as follows. First we calculate the Cholesky decomposition, L, of C_y, satisfying $LL^{\mathrm{T}} = C_y$. As mentioned in section 3.1, since C_y is banded, this can be performed in $\mathcal{O}(T)$ operations. Then we can rewrite equation 4.3 as

$$J_y = L^{-1^{\mathrm{T}}} Q L^{-1}, \qquad Q \equiv \mathbf{I} + L^{\mathrm{T}} J_y^{\mathrm{LL}} L. \tag{4.4}$$

Since L is banded (due to the bandedness of C_y) and J_y^{LL} is diagonal, it follows that Q is also banded. Therefore, its Cholesky decomposition, B, satisfying $B^{\mathrm{T}}B = Q$, can be calculated in $\mathcal{O}(T)$ time and is also banded. Using this definition and inverting equation 4.4, we obtain $AA^{\mathrm{T}} = J_y^{-1} = LB^{-1}\left(LB^{-1}\right)^{\mathrm{T}}$, from which we deduce $A = LB^{-1}$, or

$$A\tilde{\mathbf{y}} = LB^{-1}\tilde{\mathbf{y}}. \tag{4.5}$$

The calculation of L and B can be performed before running the HMC chain. Then at each step we need to perform equation 4.5. As described in the final paragraph of section 3.1, calculating $B^{-1}\tilde{\mathbf{y}}$ and the multiplication of the resulting vector by L require only $\mathcal{O}(T)$ elementary operations due to the bandedness of B and L.

Figure 8 shows the spike trains, as well as the corresponding true inputs and MAP and posterior mean estimates. The closeness of the posterior mean to the MAP (the L_2 norm of their difference is only about 9% of the L_2 norm of the MAP) is an indication of the accuracy of the Laplace approximation in this case.

5 Other Applications: Estimation of Nonmarginal Quantities

So far we have focused on using the MCMC samples to estimate $E(\mathbf{x} \mid \boldsymbol{r})$ or the posterior covariance. Both quantities involve separate averaging over the marginal distribution of single components or pairs of components of $\mathbf{x}$. However, since MCMC provides samples from the joint distribution $p(\mathbf{x} \mid \boldsymbol{r})$, we can also calculate quantities that cannot be reduced to averages over one- or two-dimensional marginal distributions and involve the whole joint distribution $p(\mathbf{x} \mid \boldsymbol{r})$. We consider two examples below.

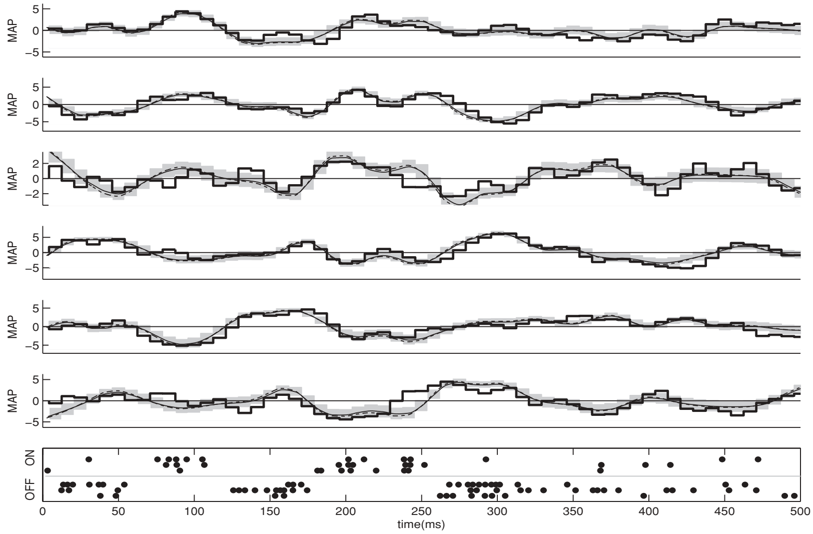

Figure 8: The top six panels show the posterior mean (solid curve) and the MAP (dashed curve) estimates of the stimulus input to three pairs of ON and OFF RGCs given their spike trains from multi-electrode array recordings. The GLM parameters used in this example were fit to data from the same recordings (see Pillow et al., 2008, for the full description of the fit GLM parameters). The jagged black traces are the actual inputs. The bottom panel shows the recorded spike trains. The posterior means were estimated using an HMC chain with 15,000 samples (after the initial 3750 samples were burned). The gray error bars around the posterior means (solid curves) represent its posterior marginal standard deviations, which were estimated using the MCMC itself (the error bars for the MAP, e.g., based on the Hessian, would not be distinguishable in this figure, and are not shown). The closeness of the posterior mean to the MAP is an indication of the accuracy of the Laplace approximation. (This decoding example also appeared briefly in Paninski et al., 2010; see also Pillow et al., 2011, the companion article.)

5.1 Posterior Statistics of Crossing Times. One important example of these nonmarginal computations involves the statistics (e.g., mean and variance) of some crossing time for the time series $\mathbf{x}$ (e.g., the time that x_t first crosses some threshold value). (First-passage time computations are especially important, for example, in the context of integrate-and-fire-based neural encoding models in Paninski, Iyengar, Kass, & Brown, 2008.) Smith et al. (2004) proposed a hidden state-space model that provides a dynamical description for the learning process of an animal in a task learning experiment (with binary responses) and yields suitable statistical indicators for establishing the occurrence of learning or determining the "learning trial."

In the proposed model, the state variable, x_t, evolves according to a gaussian random walk from trial to trial (labeled by t), and the probability of a correct response on every trial, q_t, is given by a logistic function of the corresponding state variable, x_t. Given the observation of the responses in all trials, the hidden state variable trajectory can be inferred. Smith et al. (2007) carried out this inference in Bayesian fashion by using Gibbs sampling from the posterior distribution over the state variable time series and the model parameters conditioned on the observed responses. There, the learning trial was defined as the first trial after which the ideal (Bayesian) observer can state with 95% confidence that the animal will perform better than chance. More mathematically, using the MCMC samples (using the winBUGS package), they obtained the sequence of the lower 95% confidence bounds for q_t for all t's (for each t, this bound depends on only the one-dimensional marginal distribution of q_t). The authors defined the learning trial as the t for which the value of this lower confidence bound crosses the probability value corresponding to chance performance and stays above it in all the following trials.

However, it is reasonable to consider several alternative definitions of learning trial in this setting. One plausible approach is to define the learning trial, t_L, in terms of certain passage times of q_t, for example, the trial in which q_t first exceeds the chance level and does not become smaller than this value at later trials. In this definition, t_L is a random variable whose value is not known by the ideal observer with certainty, and its statistics is determined by the full joint posterior distribution and cannot be obtained from its marginals. The posterior mean of t_L provides an estimate for this quantity and its posterior variance, an estimate of its uncertainty. These quantities involve nonlinear expectations over the full joint posterior distribution of $\{x_t\}$ and can be estimated by the MCMC samples from that distribution.

Figure 9 shows a simulated example in which we used our MCMC methods to decode the crossing times of the input to a Poisson neuron based on the observation of its spike train. The neuron's rate was given by $\lambda(t) = \exp(x_t + b)$, and the threshold corresponded to a value $x_t = x_0$. The hidden process x_t was assumed to evolve according to a gaussian AR(1) process, as in Smith et al. (2007). After a spike train was observed, samples from the posterior distribution of x_t were obtained by an HMC chain. To estimate the first and the last times that x_t crosses x_0 from below, we calculate these times for each MCMC sample, obtaining samples from the posterior distribution of these times. Then we calculate their sample mean to estimate when learning occurs. Figure 9 shows the full histograms of these passage times, emphasizing that these statistics are not fully determined by a single observation of the spike train.

As a side note, to obtain a comparison between the performance of the Gibbs-based winBUGS package employed in Smith et al. (2007) versus the HMC chain used here, we simulated a Gibbs chain for $y(t)$ on the same posterior distribution. The estimated correlation time of the Gibbs chain was

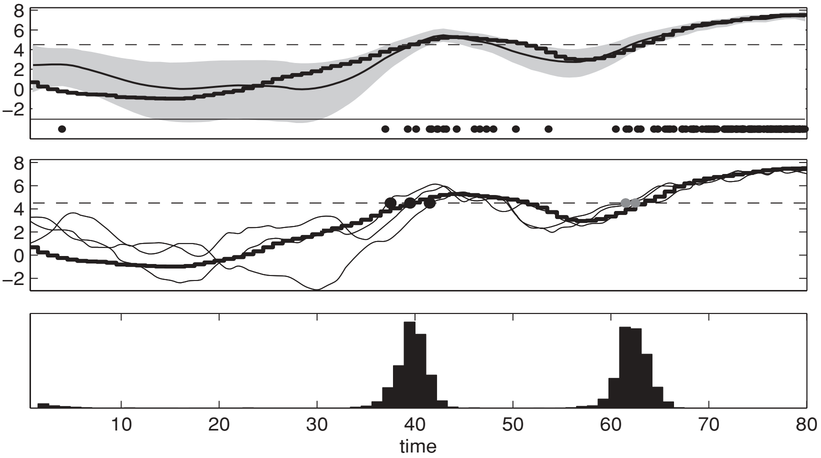

Figure 9: Estimation of threshold crossing times using MCMC sampling. The spike trains are generated by an inhomogeneous Poisson process with a rate $\lambda(t) = \exp(x_t + b)$ that depends on a changing hidden variable x_t (times are in arbitrary units). Having observed a particular spike train (bottom row of the top panel), the goal is to estimate the first or the last time that x_t crosses a threshold from below. The top and the middle plots show the true x_t (black jagged lines) and the threshold (the dashed horizontal lines). The top plot also shows the posterior marginal median for x_t (curvy line) given the observed spike train, and its corresponding posterior marginal 90% confidence interval (shaded area). Smith et al. (2007) used these marginal statistics to estimate the crossing times. However, a more systematic way of estimating these times is to directly use their (nonmarginal) posterior statistics. The middle plot also shows three posterior samples of x_t (curvy lines) obtained using an HMC Markov chain. The first and last crossing times are well defined for each of these three curves and are marked by black and gray dots, respectively. For each MCMC sample curve, we calculated these crossing times and then tabulated the statistics of these times across all samples. The bottom panel shows the MCMC-based posterior histograms of these crossing times thus obtained. The two separated peaks correspond to the first and the last crossing times. The posterior mean and variance of the crossing times can then be calculated from these histograms.

≈ 130—that is, Gibbs mixes a hundred times slower than the HMC chain here due to the nonnegligible temporal correlations in x_t (see Figure 9; recall Figure 3). In addition, due to the state-space nature of the prior on x_t here, the Hessian of the log posterior on $\mathbf{x}$ is tridiagonal, and therefore the HMC update requires just $\mathcal{O}(T)$ time, just like a full Gibbs sweep.

5.2 Mutual Information. Our second example is the calculation of the mutual information. Estimates of information transfer rates of neural systems, and the mutual information between the stimulus and response of some neural population, are essential in the study of the neural encoding and decoding problems (Bialek, Rieke, de Ruyter van Steveninck, & Warland, 1991; Warland, Reinagel, & Meister, 1997; Barbieri et al., 2004). Estimating this quantity is known to be often computationally quite difficult, particularly for high-dimensional stimuli and responses (Paninski, 2003). Pillow et al. (2011), in the companion article, present an easy and efficient method for calculating the mutual information for neurons modeled by the GLM, equations 2.3 and 2.4, based on the Laplace approximation, equation 2.8. As discussed above, this approximation is expected to hold in the case of gaussian priors, in a broad region of the GLM parameter space. Our goal here is to verify this intuition by comparing the Laplace approximation for the mutual information with an exact direct estimation using MCMC integration. As we will see, the main difficulty in using MCMC to estimate the mutual information lies in the fact that we can calculate $p(\mathbf{x} \mid \boldsymbol{r})$ only up to an unknown normalization constant. Estimating this unknown constant turns out to be tricky in that naive methods for calculating it lead to large sampling errors. Below, we use an efficient low-error method, known as bridge sampling, for estimating this constant.

The mutual information is by definition equal to the average reduction in the uncertainty regarding the stimulus (i.e., the entropy, H, of the distribution over the stimulus) of an ideal observer having access to the spike trains of the RGC, compared to its prior state of knowledge about the stimulus:

$$\begin{aligned} I[\mathbf{x}; \boldsymbol{r}] &= H[\mathbf{x}] - E(H[\mathbf{x} \mid \boldsymbol{r}]) \\ &\equiv -\int p(\mathbf{x}) \log p(\mathbf{x})\, d\mathbf{x} + \left\langle \int p(\mathbf{x} \mid \boldsymbol{r}) \log p(\mathbf{x} \mid \boldsymbol{r})\, d\mathbf{x} \right\rangle_{p(\boldsymbol{r})}. \end{aligned} \tag{5.1}$$

Here, $p(\boldsymbol{r})$ is given by equation 2.6, and the posterior probability $p(\mathbf{x} \mid \boldsymbol{r})$ is given by Bayes' rule, equation 2.5. The logarithms are assumed to be in base 2, so that information is measured in bits. We consider gaussian priors given by equation 2.10, for which we can compute the entropy $H[\mathbf{x}]$ explicitly:

$$H[\mathbf{x}] = \frac{d}{2} \log 2\pi e + \frac{1}{2} \log |\mathcal{C}|. \tag{5.2}$$

Thus, the real problem is to evaluate the second term in equation 5.1. The integral involved in the definition of $H[\mathbf{x} \mid \boldsymbol{r}]$ is in general hard to evaluate. One approach, which is computationally very fast, is to use the Laplace approximation, equation 2.8, if it is justified; we took this approach in Pillow et al. (2011), the companion article. In that case, from equation 2.8, we obtain

$$I[\mathbf{x}; \boldsymbol{r}] \approx \left\langle \frac{1}{2} \log |\, \mathcal{C} \cdot J(\boldsymbol{r})\, | \right\rangle_{p(\boldsymbol{r})} \equiv \langle I_L(\boldsymbol{r}) \rangle_{p(\boldsymbol{r})} = I_L, \tag{5.3}$$

where $J(\boldsymbol{r})$ is the Hessian, equation 2.9.

More generally, we can use the MCMC method developed in section 3 to estimate $H[\mathbf{x} \mid \boldsymbol{r}]$ directly. The integral involved in $H[\mathbf{x} \mid \boldsymbol{r}]$, equation 5.1 (before averaging over $p(\boldsymbol{r})$), has the form

$$E\left(g(\mathbf{x}) \mid \boldsymbol{r}\right) = \int g(\mathbf{x}) p(\mathbf{x} \mid \boldsymbol{r}) d\mathbf{x}, \tag{5.4}$$

that is, one representing the posterior expectation of a function $g(\mathbf{x})$. If we could evaluate $g(\mathbf{x})$ for arbitrary $\mathbf{x}$, we could evaluate this expectation by the MCMC method, via equation 3.2. As we mentioned above, the difficulty lies in that in general, we can evaluate only an unnormalized version of the posterior distribution, and thus $g(\mathbf{x}) = -\log p(\mathbf{x} \mid \boldsymbol{r})$, only up to an additive constant. Suppose we can evaluate

$$q(\mathbf{x} \mid \boldsymbol{r}) \equiv Z(\boldsymbol{r}) p(\mathbf{x} \mid \boldsymbol{r}), \tag{5.5}$$

for some $Z(\boldsymbol{r})$ at any arbitrary $\mathbf{x}$. Then $H[\mathbf{x} \mid \boldsymbol{r}]$ can be rewritten as

$$H[\mathbf{x} \mid \boldsymbol{r}] = \log Z(\boldsymbol{r}) - \left\langle \log q(\mathbf{x} \mid \boldsymbol{r}) \right\rangle_{p(\mathbf{x}|\boldsymbol{r})}. \tag{5.6}$$

From the normalization condition for $p(\mathbf{x} \mid \boldsymbol{r})$, $Z(\boldsymbol{r})$ is given by

$$Z(\boldsymbol{r}) = \int q(\mathbf{x} \mid \boldsymbol{r})\, d\mathbf{x}. \tag{5.7}$$

The main difficulty in calculating the mutual information lies in estimating $Z(\boldsymbol{r})$ (for a discussion of the difficulties involved in estimating normalization constants and marginal probabilities see Meng & Wong, 1996; the discussion of the paper by Newton & Raftery, 1994; and Neal, 2008). By contrast, the second term in equation 5.6 already has the form equation 5.4 (with $q(\mathbf{x} \mid \boldsymbol{r})$ replacing $g(\mathbf{x})$) and can be estimated using equation 3.2. In the following, we introduce an efficient method for estimating $Z(\boldsymbol{r})$.

As noted above, if in equations 5.6 and 5.7, we replace $q(\mathbf{x} \mid \boldsymbol{r})$ with the Laplace approximation,

$$q_{L}(\mathbf{x} \mid \boldsymbol{r}) \equiv e^{-\frac{1}{2}(\mathbf{x}-\mathbf{x}_{\mathrm{MAP}})^{\mathrm{T}} J\, (\mathbf{x}-\mathbf{x}_{\mathrm{MAP}}) - \mathcal{L}_0} \equiv Z_{L}(\boldsymbol{r}) p_{L}(\mathbf{x} \mid \boldsymbol{r}), \tag{5.8}$$

we obtain the result in equation 5.3, as a first approximation to the mutual information. Here we defined

$$\mathcal{L}_0 \equiv -\ln q(\mathbf{x}_{\mathrm{MAP}} \mid \boldsymbol{r}), \tag{5.9}$$

and from the normalization condition for $p_{L}(\mathbf{x} \mid \boldsymbol{r})$, we must have

$$Z_{L}(\boldsymbol{r}) = \int q_{L}(\mathbf{x} \mid \boldsymbol{r})\, d\mathbf{x} = \sqrt{(2\pi)^{d} \mid J\,(\boldsymbol{r}) \mid^{-1}} e^{-\mathcal{L}_0}. \tag{5.10}$$

We included the constant $\mathcal{L}_0$ in the exponent in equation 5.8 so that when the Laplace approximation is accurate, we have $\log q_{_L}(\mathbf{x} \mid \boldsymbol{r}) \approx \log q(\mathbf{x} \mid \boldsymbol{r})$ and $\log Z(\boldsymbol{r}) \approx \log Z_{_L}(\boldsymbol{r})$.

We now write the exact mutual information as $I[\mathbf{x}; \boldsymbol{r}] = \langle I(\boldsymbol{r}) \rangle_{p(\boldsymbol{r})}$, and write $I(\boldsymbol{r}) \equiv H[\mathbf{x}] - H[\mathbf{x} \mid \boldsymbol{r}]$ as the Laplace approximation for it plus a correction,

$$I(\boldsymbol{r}) = I_L(\boldsymbol{r}) + \delta I(\boldsymbol{r}) = \frac{1}{2} \log |\, \mathcal{C} \cdot J(\boldsymbol{r}) \,| + \delta I(\boldsymbol{r}), \tag{5.11}$$

where

$$\delta I(\boldsymbol{r}) \equiv -(H[\mathbf{x} \mid \boldsymbol{r}] - H_L[\mathbf{x} \mid \boldsymbol{r}]). \tag{5.12}$$

Using the general formula 5.6 for both the true distribution, equation 5.5, and its gaussian approximation, equation 5.8, we obtain

$$\begin{aligned}\delta I(\boldsymbol{r}) &= -(H[\mathbf{x} \mid \boldsymbol{r}] - H_L[\mathbf{x} \mid \boldsymbol{r}]) \equiv \big\langle \log q(\mathbf{x} \mid \boldsymbol{r}) \big\rangle_{p(\mathbf{x}|\boldsymbol{r})} - \big\langle \log q_{_L}(\mathbf{x} \mid \boldsymbol{r}) \big\rangle_{p_{_L}(\mathbf{x}|\boldsymbol{r})} \\ &\quad - \log Z(\boldsymbol{r}) + \log Z_{_L}(\boldsymbol{r}) \\ &= \big\langle \log q(\mathbf{x} \mid \boldsymbol{r}) \big\rangle_{p(\mathbf{x}|\boldsymbol{r})} - \big\langle \log q_{_L}(\mathbf{x} \mid \boldsymbol{r}) \big\rangle_{p_{_L}(\mathbf{x}|\boldsymbol{r})} - \log \eta,\end{aligned} \tag{5.13}$$

with $\eta \equiv Z(\boldsymbol{r})/Z_{_L}(\boldsymbol{r})$. Thus, after averaging over $p(\boldsymbol{r})$, $\delta I(\boldsymbol{r})$, calculated using equation 5.13, gives the correction to the Laplace approximation for the mutual information, equation 5.3. When the Laplace approximation is justified, this correction will be small (even before averaging over $p(\boldsymbol{r})$). Also, note that in that case $\eta \approx 1$, and the last term in equation 5.13 is small on its own.

The second term in equation 5.13 is readily evaluated;

$$-\big\langle \log q_{_L}(\mathbf{x} \mid \boldsymbol{r}) \big\rangle_{p_{_L}(\mathbf{x}|\boldsymbol{r})} = \frac{d}{2 \ln 2} + \frac{\mathcal{L}_0}{\ln 2}, \tag{5.14}$$

and the first term can be evaluated using the MCMC via equation 3.2. To evaluate the third term, we use the following trick. For any well-behaved function $\alpha(\mathbf{x})$, we have

$$\eta = \frac{Z(\boldsymbol{r})}{Z_L(\boldsymbol{r})} = \frac{Z(\boldsymbol{r})}{Z_L(\boldsymbol{r})} \frac{\int p(\mathbf{x} \mid \boldsymbol{r}) \alpha(\mathbf{x}) p_{_L}(\mathbf{x} \mid \boldsymbol{r})\, d\mathbf{x}}{\int p_{_L}(\mathbf{x} \mid \boldsymbol{r}) \alpha(\mathbf{x}) p(\mathbf{x} \mid \boldsymbol{r})\, d\mathbf{x}} = \frac{\langle q(\mathbf{x} \mid \boldsymbol{r}) \alpha(\mathbf{x}) \rangle_{p_{_L}(\mathbf{x}|\boldsymbol{r})}}{\langle q_{_L}(\mathbf{x} \mid \boldsymbol{r}) \alpha(\mathbf{x}) \rangle_{p(\mathbf{x}|\boldsymbol{r})}}. \tag{5.15}$$

Using this formula, we can estimate η by estimating the numerator and denominator on the right-hand side according to equation 3.2, with samples drawn from $p(\mathbf{x} \mid \boldsymbol{r})$ and $p_{_L}(\mathbf{x} \mid \boldsymbol{r})$, respectively, for example, by MCMC.

However, as we have access only to finitely many samples from each distribution, care must be taken in the choice of the function α to avoid large estimation errors. For example, if the support of $p(\mathbf{x} \mid \boldsymbol{r})$ and $p_{_L}(\mathbf{x} \mid \boldsymbol{r})$ has a small overlap, $\alpha(\mathbf{x})$ has to be chosen such that it amplifies the contribution from the region of overlap of the two distributions, thus acting as a bridge connecting the two supports. Otherwise (e.g., if $\alpha(\mathbf{x})$ is a constant) both the numerator and denominators in equation 5.15 can be very small in such a case, leading to an almost indeterminate ratio with large random error.[14] A method of evaluating η using equation 5.15 by employing an optimal $\alpha(\mathbf{x})$, was originally developed by Bennett (1976) and was further refined by Meng and Wong (1996); it is referred to as bridge sampling for the above reason. These authors have shown that the asymptotically optimal (for large number of samples from each distribution) choice of $\alpha(\mathbf{x})$ is

$$\alpha(\mathbf{x}) \propto \frac{1}{s_1 p(\mathbf{x} \mid \boldsymbol{r}) + s_2 p_L(\mathbf{x} \mid \boldsymbol{r})} \propto \frac{1}{s_1 q(\mathbf{x} \mid \boldsymbol{r}) + \eta s_2 q_L(\mathbf{x} \mid \boldsymbol{r})}, \tag{5.16}$$

where $s_i = N_i/(N_1 + N_2)$ $(i = 1, 2)$ and $N_{1,2}$ are the number of samples drawn from $p(\mathbf{x} \mid \boldsymbol{r})$ and $p_L(\mathbf{x} \mid \boldsymbol{r})$, respectively. As this choice for $\alpha(\mathbf{x})$ itself depends on η, it suggests an iterative solution, where we substitute equation 5.16 with the current η into equation 5.15 to obtain the next iteration:

$$\hat{\eta}^{(t+1)} = \frac{\frac{1}{N_2}\sum_{j=1}^{N_2} \frac{q(\mathbf{x}_{2j}|\boldsymbol{r})}{s_1 q(\mathbf{x}_{2j}|\boldsymbol{r}) + \hat{\eta}^{(t)} s_2 q_L(\mathbf{x}_{2j}|\boldsymbol{r})}}{\frac{1}{N_1}\sum_{j=1}^{N_1} \frac{q_L(\mathbf{x}_{1j}|\boldsymbol{r})}{s_1 q(\mathbf{x}_{1j}|\boldsymbol{r}) + \hat{\eta}^{(t)} s_2 q_L(\mathbf{x}_{1j}|\boldsymbol{r})}} = \frac{\frac{1}{N_2}\sum_{j=1}^{N_2} \frac{l_{2j}}{s_1 l_{2j} + \hat{\eta}^{(t)} s_2}}{\frac{1}{N_1}\sum_{j=1}^{N_1} \frac{1}{s_1 l_{1j} + \hat{\eta}^{(t)} s_2}}, \tag{5.17}$$

where $\mathbf{x}_{ij}$ $(i = 1, 2)$ are samples drawn from $p(\mathbf{x} \mid \boldsymbol{r})$ and $p_L(\mathbf{x} \mid \boldsymbol{r})$, respectively, and $l_{ij} \equiv q(\mathbf{x}_{ij} \mid \boldsymbol{r})/q_{_L}(\mathbf{x}_{ij} \mid \boldsymbol{r})$. Since we expect $Z \approx Z_{_L}$, we take $\hat{\eta}^{(0)} = 1$. In our calculations, we stopped the bridge sampling iterations when $\log \frac{\hat{\eta}^{(t+1)}}{\hat{\eta}^{(t)}} < 0.001d$, where d is the stimulus dimension.

Figure 10 shows a plot of $I_{_L}$ and δI per stimulus dimension, calculated as described above,[15] as a function of the standard deviation of the filtered stimulus input (which, for the white noise stimulus, is the contrast, c, times the magnitude of $k_i(t)$). $I_{_L}$ grows as $c\|k_i\|$ grows, but δI does not change significantly and remains small. Thus, the Laplace approximation for the

---

[14] For a similar reason, a "brute force" method for computing Z, such as a simple Monte Carlo integration of the high-dimensional distribution $q(\mathbf{x} \mid \boldsymbol{r})$, gives rise to an estimate with slow convergence and a large error (Meng & Wong, 1996).

[15] In principle, according to equations 5.1 to 5.3, one has to average $I_L(\boldsymbol{r})$ and $\delta I(\boldsymbol{r})$ over the marginal response distribution $p(\boldsymbol{r})$. But due to the intensive nature of $I_{_L}(\boldsymbol{r})/d$ and $\delta I(\boldsymbol{r})/d$ and the large d, they depended on the specific realization of $\boldsymbol{r}$ only weakly, and they were to a good degree self-averaging, so that averaging over $p(\boldsymbol{r})$ would not considerably alter the plot of Figure 10. A similar argument appeared in Strong, Koberle, de Ruyter van Steveninck, & Bialek, 1998.

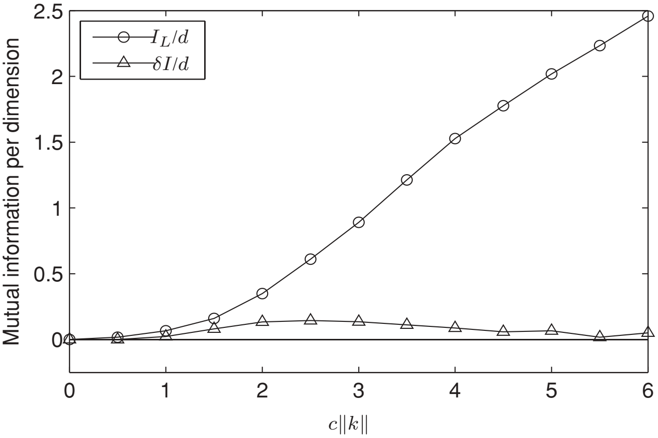

Figure 10: Comparison of Laplace approximation to mutual information per stimulus dimension, I_L/d, and the correction, $\delta I/d$ (see equation 5.11), based on the MCMC estimate of the true value for a pair of ON and OFF RGCs, as a function of the magnitude of the filtered stimulus input $c\|k\|$, where c is the contrast and $\|k\|$ is the norm of the stimulus filter. The computationally inexpensive Laplace approximation for the mutual information is accurate for moderately strong stimulus filters, which give rise to sharp likelihoods. Furthermore, at $c\|k\| = 0$, the likelihood has no dependence on $\mathbf{x}$, and the posterior is equal to the gaussian prior, for which the Laplace approximation is exact. Thus, for very small $\|k\|$ also, I_L becomes exact and the error, δI, has a maximum around $c\|k\| \approx 2.5$.

mutual information is accurate for moderately large $c\|k_i\|$. Furthermore, for vanishing $c\|k_i\|$, the posterior is equal to the gaussian prior in this case, and this approximation is exact. Therefore, the error δI has a maximum at a finite value of the stimulus filter, away from which the Laplace approximation is accurate. For comparison with our real spike data example presented in section 4 and Figure 8, we note that in that case, the standard deviation of the filtered stimulus to different cells was in the range $c\|k\| \sim 3 \pm 1$, depending on the cell, and the Laplace approximation did indeed provide an accurate approximation for the mutual information, with $\delta I/I_L = 0.09$.

6 Effect of Uncertainty in the Model Parameters

In the previous sections we assumed the values of the parameters involved in the GLM likelihood, equations 2.3 and 2.4, were known exactly. Of course, in reality these parameters themselves are obtained by fitting the GLM to experimental data and are thus known only with a finite accuracy. In this section, we investigate the effect of uncertainty in the GLM parameters θ (see section 2) on the posterior mean estimate for the stimulus. We represent this uncertainty by a probability distribution, $p(\theta)$. In the presence of parameter uncertainty, the posterior mean of the stimulus, $\mathbf{x}$, is modified to

$$E(\mathbf{x} \mid \boldsymbol{r}) = \int E(\mathbf{x} \mid \boldsymbol{r}, \theta) p(\theta)\, d\theta = \int \mathbf{x}\, p(\mathbf{x} \mid \boldsymbol{r}, \theta) p(\theta)\, d\mathbf{x}\, d\theta. \tag{6.1}$$

(In this section, unlike in sections 3 to 5, we write θ explicitly when a distribution is conditioned on it. When there is no θ in the argument of the distribution, it means it has been marginalized.) We assume the uncertainty in the parameters is small enough that a Laplace approximation for $p(\theta)$ applies, that is, it can be taken to be gaussian with mean θ_{ML} and a small covariance $I^{-1}(\theta_{\mathrm{ML}})$. Here, θ_{ML} is the maximum likelihood fit to data, and $I^{-1}(\theta_{\mathrm{ML}})$ is the Hessian of the negative log likelihood (as a function of GLM parameters, given the experimental data) at θ_{ML}. For simplicity we assume the GLM nonlinearity (see equation 2.3) is exponential: $f(u) = \exp(u)$. We also assume that the prior stimulus ensemble is gaussian, with probability distribution described by equation 2.10.

We would like to understand how the uncertainty in θ will affect the posterior estimate. This uncertainty broadens the likelihood (as a function of $\mathbf{x}$), and therefore we expect that as it increases, the posterior estimate $E(\mathbf{x} \mid \boldsymbol{r})$ will move toward the prior mean (in our case, zero). Intuitively, this is because as the Bayesian decoder's knowledge of the encoding mechanism (represented by the parameters θ) decreases, it discounts the information that the observed spike train, $\boldsymbol{r}$, carries about the stimulus and instead relies more strongly on its prior information. To verify this intuition analytically, we consider the case where $E(\mathbf{x} \mid \boldsymbol{r}, \theta) \approx \mathbf{x}_{\mathrm{MAP}}(\boldsymbol{r}, \theta)$ (e.g., as we saw in the previous section, the Laplace approximation for the posterior stimulus distribution is often quite adequate in the case of gaussian priors, and this approximation therefore holds in that case). Assuming this, we can replace $E(\mathbf{x} \mid \boldsymbol{r}, \theta)$ with $\mathbf{x}_{\mathrm{MAP}}(\boldsymbol{r}, \theta)$ in equation 6.1, and obtain

$$E(\mathbf{x} \mid \boldsymbol{r}) \approx \int \mathbf{x}_{\mathrm{MAP}}(\boldsymbol{r}, \theta) p(\theta)\, d\theta. \tag{6.2}$$

In the following, we drop $\boldsymbol{r}$ from the arguments of $\mathbf{x}_{\mathrm{MAP}}$ when it is understood. We denote the average over $p(\theta)$ in equation 6.2 by $\langle \mathbf{x}_{\mathrm{MAP}} \rangle_\theta$.

Using Bayes's rule in the form $\log p(\mathbf{x} \mid \boldsymbol{r}) = \log p(\mathbf{x}) + \log p(\boldsymbol{r} \mid \mathbf{x}) +$ const., with equation 2.10, and equations 2.3 and 2.4 with exponential nonlinearity, we obtain

$$-\log p(\mathbf{x} \mid \boldsymbol{r}, \theta) = \frac{1}{2}\mathbf{x}^{\mathrm{T}} C^{-1}\mathbf{x} + \sum_i \Big[-\boldsymbol{r}_i^{\mathrm{T}} \cdot \big(K_i \cdot \mathbf{x} + b_i + \mathcal{H}_{ij} \cdot \boldsymbol{r}_j \big) \\ + \int e^{K_i \cdot \mathbf{x} + b_i + \sum_j \mathcal{H}_{ij} \cdot \boldsymbol{r}_j} dt \Big], \tag{6.3}$$

up to an additive constant. Here, $\mathcal{C}$ is the covariance of the gaussian prior, equation 2.10, and $\theta = \{b_i, K_i, \mathcal{H}_{ij}\}$ are the GLM parameters introduced in section 2. The MAP satisfies

$$\frac{\partial \log p(\mathbf{x}_{\text{MAP}} \mid \boldsymbol{r})}{\partial \mathbf{x}} = 0, \tag{6.4}$$

which yields the equation

$$\mathcal{C}^{-1}\mathbf{x}_{\text{MAP}}(\theta) = \sum_i K_i^{\mathrm{T}} \cdot \left[\boldsymbol{r}_i - e^{K_i \cdot \mathbf{x}_{\text{MAP}}(\theta) + b_i + \sum_j \mathcal{H}_{ij} \cdot \boldsymbol{r}_j}\right]. \tag{6.5}$$

When the contrast or the stimulus filter is small (corresponding to the regime of low signal-to-noise ratio), the exponential can be expanded to first order in $\mathbf{x}_{\text{MAP}}(\theta)$, yielding the linear equation (a similar expansion also appeared in Pillow et al., 2011, the companion article),

$$\mathcal{A}(\theta)\mathbf{x}_{\text{MAP}}(\theta) = \mathcal{B}(\theta), \tag{6.6}$$

where we defined

$$\mathcal{A}(\theta) \equiv \mathcal{C}^{-1} + \sum_i K_i^{\mathrm{T}} \mathcal{S}_i K_i, \tag{6.7}$$

$$\mathcal{B}(\theta) = \sum_i K_i^{\mathrm{T}} \cdot (\boldsymbol{r}_i - \boldsymbol{S}_i), \tag{6.8}$$

and

$$\boldsymbol{S}_i \equiv e^{b_i + \sum_j \mathcal{H}_{ij} \boldsymbol{r}_j}, \tag{6.9}$$

$$\mathcal{S}_i(t_1, t_2) \equiv \boldsymbol{S}_i(t_1)\delta(t_1 - t_2). \tag{6.10}$$

Notice that $\mathcal{A}(\theta)$ is the Hessian of the negative log posterior, equation 6.3, at $\mathbf{x} = 0$. Assuming the matrix $\mathcal{A}(\theta)$ is invertible,[16] we then obtain

$$\mathbf{x}_{\text{MAP}}(\theta) = \mathcal{A}(\theta)^{-1}\mathcal{B}(\theta). \tag{6.11}$$

We write $\theta = \theta_{\text{ML}} + \delta\theta$, where $\delta\theta$ has zero mean, and expand equation 6.11 in $\delta\theta$ up to second order to obtain

$$\mathbf{x}_{\text{MAP}}(\theta) = \mathbf{x}^{(0)}_{\text{MAP}} + \mathbf{x}^{(1)}_{\text{MAP}} + \mathbf{x}^{(2)}_{\text{MAP}} + O(\delta\theta^3), \tag{6.12}$$

---

[16] This is true when $\mathcal{C}$ is well-defined.

where

$$\mathbf{x}^{(0)}_{\text{MAP}} = \mathcal{A}_0^{-1}\mathcal{B}_0, \tag{6.13}$$

and

$$\mathbf{x}^{(1)}_{\text{MAP}} = \mathcal{A}_0^{-1}d\mathcal{B} - \mathcal{A}_0^{-1}d\mathcal{A}\cdot\mathbf{x}^{(0)}_{\text{MAP}}, \tag{6.14}$$

$$\mathbf{x}^{(2)}_{\text{MAP}} = \mathcal{A}_0^{-1}\left[d^2\mathcal{B} - d\mathcal{A}\mathcal{A}_0^{-1}d\mathcal{B} - \left(d^2\mathcal{A} - d\mathcal{A}\mathcal{A}_0^{-1}d\mathcal{A}\right)\mathbf{x}^{(0)}_{\text{MAP}}\right], \tag{6.15}$$

such that $\mathbf{x}^{(n)}_{\text{MAP}}$ is homogeneously of order n in $\delta\theta$. Here, we defined $\mathcal{A}_0 \equiv \mathcal{A}(\theta_{\text{ML}})$, $\mathcal{B}_0 \equiv \mathcal{B}(\theta_{\text{ML}})$, and $d\mathcal{A}$ and $d\mathcal{B}$ ($d^2\mathcal{A}$ and $d^2\mathcal{B}$) are the random first- (second-) order variations of $\mathcal{A}(\theta)$ and $\mathcal{B}(\theta)$ in $\delta\theta$. The first-order variations, $d\mathcal{A}$ and $d\mathcal{B}$, are thus gaussian with zero mean and a covariance determined by the covariance of θ. After averaging over θ, $\mathbf{x}^{(1)}_{\text{MAP}}$ will vanish, and we have

$$\langle\mathbf{x}_{\text{MAP}}\rangle_\theta = \mathbf{x}^{(0)}_{\text{MAP}} + \langle\mathbf{x}^{(2)}_{\text{MAP}}\rangle_\theta. \tag{6.16}$$

To gain some intuition, we now set out to evaluate $\mathbf{x}^{(2)}_{\text{MAP}}$ in the regime of small baseline firing rates, so that $\mathcal{S}_i^0 \equiv \mathcal{S}_i(\theta_{\text{ML}})$ are small, and we also assume we can neglect the uncertainty of the baseline firing rates and the postspike feedback filters (i.e., we set $\delta b_i = \delta\mathcal{H}_{ij} = 0$). In this case, $d^2\mathcal{B} = 0$, and ignoring terms beyond the leading order in $\mathcal{S}_i^0$, we take $\mathcal{A}_0^{-1} \approx \mathcal{C}$, and obtain

$$\mathbf{x}^{(2)}_{\text{MAP}} \approx -\mathcal{A}_0^{-1}\left(d\mathcal{A}\mathcal{A}_0^{-1}d\mathcal{B} + d^2\mathcal{A}\mathcal{A}_0^{-1}\mathcal{B}_0\right), \tag{6.17}$$

$$\begin{aligned}&\approx -\mathcal{C}\sum_{ij}\left[\overline{K}_i^{\mathrm{T}}\mathcal{S}_0^i\delta K_i\mathcal{C}\delta K_j^{\mathrm{T}} + \delta K_i^{\mathrm{T}}\mathcal{S}_i^0\overline{K}_i\mathcal{C}\delta K_j^{\mathrm{T}} + \delta K_i^{\mathrm{T}}\mathcal{S}_i^0\delta K_i\mathcal{C}\overline{K}_j^{\mathrm{T}}\right]\\ &\quad\cdot\left(\boldsymbol{r}_j - \mathcal{S}_j^0\right)\\ &= -\mathcal{C}^2\sum_{ij}\left[\overline{K}_i^{\mathrm{T}}\mathcal{S}_i^0\delta K_i\delta K_j^{\mathrm{T}} + \delta K_i^{\mathrm{T}}\mathcal{S}_i^0\overline{K}_i\delta K_j^{\mathrm{T}} + \delta K_i^{\mathrm{T}}\mathcal{S}_i^0\delta K_i\overline{K}_j^{\mathrm{T}}\right]\\ &\quad\cdot\left(\boldsymbol{r}_j - \mathcal{S}_j^0\right).\end{aligned} \tag{6.18}$$

Here, we denoted the maximum likelihood fit for the stimulus filters by $\overline{K}_i$, and in deriving the second line, we used $d\mathcal{A} = \sum_i \delta K_i^{\mathrm{T}}\mathcal{S}_i^0\overline{K}_i + \overline{K}_i^{\mathrm{T}}\mathcal{S}_i^0\delta K_i$, $d^2\mathcal{A} = \sum_i \delta K_i^{\mathrm{T}}\mathcal{S}_i^0\delta K_i$, and $d\mathcal{B} = \sum_j \delta K_j(\boldsymbol{r}_j - \mathcal{S}_j^0)$, and in the last line we assumed the stimulus is white, that is, $\mathcal{C} \propto \mathbf{1}$. Equation 6.18 is not very enlightening, so we look at the special case where $\delta K_i = \alpha\overline{K}_i$, and α is a noisy gaussian scalar with zero mean (this arises, for example, in the case of delta function kernels, as in the example of the last section—or more generally when only the overall scale of K_i is uncertain). Replacing for δK_i and using equation 6.13 with $\mathcal{A}_0^{-1} \approx \mathcal{C} = c^2\mathbf{1}$ to write $c^2 \sum_j K_j^{\mathrm{T}}(\boldsymbol{r}_j - \mathcal{S}_j^0) = \mathbf{x}^{(0)}_{\mathrm{MAP}}$, for this case we obtain

$$\left\langle \mathbf{x}^{(2)}_{\mathrm{MAP}} \right\rangle_\theta \approx -3\langle \alpha^2 \rangle_\theta c^2 \sum_i \overline{K}_i^{\mathrm{T}} \mathcal{S}_i^0 \overline{K}_i \mathbf{x}^{(0)}_{\mathrm{MAP}}. \tag{6.19}$$

Therefore, to the first nonvanishing order, the change in the L^2 norm of the estimate is

$$\begin{aligned} \| E(\mathbf{x} \mid \boldsymbol{r}) \|_2^2 - \| E(\mathbf{x} \mid \boldsymbol{r}, \theta_{\mathrm{ML}}) \|_2^2 &\approx \| \langle \mathbf{x}_{\mathrm{MAP}} \rangle_\theta \|_2^2 - \left\| \mathbf{x}^{(0)}_{\mathrm{MAP}} \right\|_2^2 \\ &\approx 2\mathbf{x}^{(0)^{\mathrm{T}}}_{\mathrm{MAP}} \cdot \left\langle \mathbf{x}^{(2)}_{\mathrm{MAP}} \right\rangle_\theta = -6\langle \alpha^2 \rangle_\theta c^2 \mathbf{x}^{(0)^{\mathrm{T}}}_{\mathrm{MAP}} \mathcal{L} \mathbf{x}^{(0)}_{\mathrm{MAP}} \le 0, \end{aligned} \tag{6.20}$$

where the inequality followed from the fact that $\mathcal{S}_i^0$, and therefore $\mathcal{L} \equiv \sum_i \overline{K}_i^{\mathrm{T}} \mathcal{S}_i^0 \overline{K}_i$ are positive definite operators. Thus, we see that, at least in the special regime that we considered, parameter uncertainty will shrink the norm of the posterior mean estimate, sending it toward the prior mean at the origin. This result is in agreement with the intuition stated above and was corroborated by our numerical results in more general parameter regimes.

Figure 11 shows a numerical plot of the norm of the posterior estimate as a function of the size of the uncertainty in K_i. Here, δK_i was not constrained to be proportional to $\overline{K}_i$. However, again, as uncertainty in model parameters increases, leading to broadening of the likelihood, the posterior mean moves toward the prior mean.

7 Discussion

Markov chain Monte Carlo allows the calculation of general, fully Bayesian posterior estimates. The main goal of this article was to survey the performance of a number of efficient MCMC algorithms in the context of model-based neural decoding of spike trains. Using these methods, we also verified and extended the results of Pillow et al. (2011) in the companion article on MAP-based decoding and information estimation, via Laplace approximation, in GLM-based neural decoding problems. Although MCMC integration is more general in this sense, it is at the same time significantly more computationally expensive than the optimization algorithms used to find the MAP. As we explained in section 2, the MAP is in general a good estimator when the Laplace approximation is accurate. The MAP also comes with natural error bars estimated through the Hessian matrix of the log posterior at MAP, equation 2.9. Furthermore, when it is valid, this approximation provides a very efficient way of estimating the mutual information

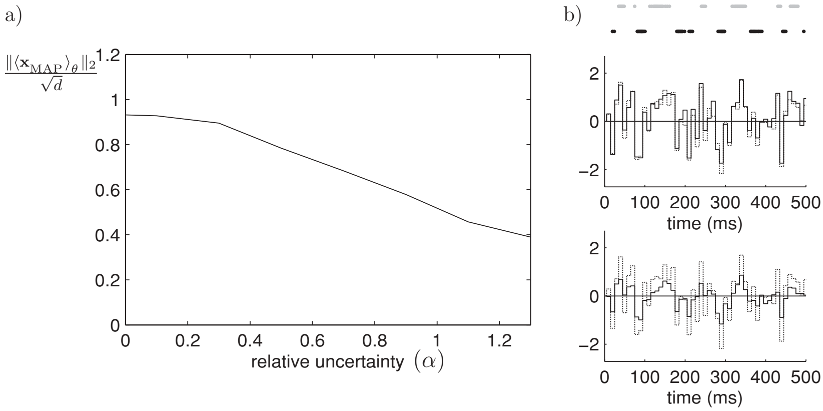

Figure 11: Effect of parameter uncertainty on the posterior estimate for gaussian white noise stimuli. (a) A plot of $\|\langle \mathbf{x}_{\mathrm{MAP}}\rangle_\theta\|_2/\sqrt{d}$ (where $d = 50$ is the stimulus dimension) versus relative uncertainty, α, in the stimulus filter $k_i(t)$. α is defined through $k_i(t) = (1 + \alpha\epsilon(t))k_i^{\mathrm{ML}}(t)$, where $\varepsilon(t)$ is a standard gaussian white noise. Unlike in section 4, $k_i^{\mathrm{ML}}(t)$ (the maximum likelihood fit for $k_i(t)$) was taken to have a time width spreading over a few stimulus frames. Furthermore, its magnitude was taken to be large enough to give rise to a sharp posterior, satisfying equation 2.8 and, thus, $E(\mathbf{x} \mid \mathbf{r}, \theta) \approx \mathbf{x}_{\mathrm{MAP}}(\mathbf{r}, \theta)$. For each value of α, 100 samples of $\varepsilon(t)$ were generated, and the MAP was decoded for each using the corresponding $k_i(t)$ and the fixed spike train. The sample average of those MAPs was taken as the estimate for $\langle \mathbf{x}_{\mathrm{MAP}}\rangle_\theta \approx E(\mathbf{x} \mid \mathbf{r})$. (b) $\langle \mathbf{x}_{\mathrm{MAP}}\rangle_\theta$ (solid trace) for $\alpha = 0$ (top plot) and $\alpha = 1$ (bottom plot) and the true stimulus (dotted trace). The main effect of the finite uncertainty is a shrinkage of the estimate toward zero (which is the mean of the prior gaussian distribution).

through equation 5.3. Thus, it is important to have a clear knowledge of when this approximation holds, since when it does, it can be exploited to dramatically reduce the computational cost of stimulus decoding or information estimation.

In section 3, we introduced the RWM, HMC, Gibbs, and hit-and-run Markov chains, all special cases of the Metropolis-Hastings algorithm. Although these methods allow sampling from general posterior distributions, regardless of the forward model, we also took advantage of the specific properties of the distributions involved in our GLM-based decoding to increase the efficiency of these chains. The ARS algorithm, which exploits the log-concavity property of the GLM likelihood and the prior distribution, was used to significantly reduce the computational cost of the one-dimensional sampling in each hit-and-run step. We took advantage of the Laplace approximation (or a regularized version of it in the flat prior case) to shape the proposal distributions to roughly match the covariance structure of the underlying distribution. Furthermore, we were able to carry this out in $\mathcal{O}(T)$ computational time (i.e., scaling only linearly with the stimulus duration, T) by exploiting the bandedness of the log-posterior Hessian in these settings. To the best of our knowledge, the use of $\mathcal{O}(T)$, Laplace-enhanced HMC in neural applications is novel. Similarly, athough the hit-and-run algorithm is well known in the statistics literature, we are unaware of any previous application of it in the context of high-dimensional neural decoding.

We mention that these chains with $\mathcal{O}(T)$, Laplace-based enhancement can also be implemented in decoding posterior distributions based on state-space models with Markovian structure; an example of such an application was presented in Figure 9, based on the state-space model used in Smith et al. (2007). However, in cases where the log-posterior turns out to be nonconcave, obtaining the Laplace approximation may be infeasible, or it may not improve the chain's mixing. Although MCMC without this enhancement is still applicable in such cases, other methods, such as sequential Monte Carlo ("particle-filtering") (Doucet, de Freitas, & Gordon, 2001; Brockwell et al., 2004; Kelly & Lee, 2004; Godsill, Doucet, & West, 2004; Shoham et al., 2005; Ergun et al., 2007; Vogelstein et al., in press; Huys and Paninski, 2009), are solely applicable in models with Markovian structure may prove to be more efficient.

It is worth noting a connection between this $\mathcal{O}(T)$ nonisotropic MCMC sampling and the Bayesian adaptive regression splines (BARS) method (DiMatteo, Genovese, & Kass, 2001; Wallstrom, Liebner, & Kass, 2007), which has become a popular tool in neursocientific applications. The BARS algorithm is a powerful nonparametric regression method designed to infer the shape of a smooth, underlying curve that has produced noisy observations. This method assumes the curve can be approximated by a spline and outputs samples from the posterior distribution of the spline knots and coefficients. Specifically, in the case of neural spike trains, it is assumed that the observed spikes, $r(t)$, are produced by an inhomogeneous Poisson process with a rate $\lambda(t) = \exp(\boldsymbol{B}(t)^{\mathrm{T}}\boldsymbol{\beta})$, where $B_i(t)$ is a cubic B-spline basis and β_i are the spline coefficients. Here, i runs from 1 to $k+2$, where k is the number of spline knots with positions τ_i; the spline basis functions, $B_i(t)$, implicitly depend on k and τ_i. Conditioned on fixed τ_i and k, the prior distribution of the spline coefficients $\boldsymbol{\beta}$ is taken to be gaussian with zero mean and inverse covariance $C_{ij}^{-1} \propto \sum_t B_i(t)B_j(t)$ (a unit information prior). Thus conditioned on fixed spline knots, the BARS model involves Poisson observations from a gaussian latent variable $\boldsymbol{\beta}$; this is directly analogous to our GLM model with gaussian stimuli, $\mathbf{x}$, with $\boldsymbol{\beta}$ and $\boldsymbol{B}(t)$ replacing $\mathbf{x}$ and K in the analogy, respectively. In particular, sampling from the posterior distribution of the a priori gaussian $\boldsymbol{\beta}$ (given τ_i and k) is very similar to sampling from the posterior over the gaussian stimulus, $\mathbf{x}$, in our examples in this article. Furthermore, to obtain conditional samples of $\boldsymbol{\beta}$, the BARS code uses an RWM chain (Wallstrom et al., 2007), which, as in equations 3.8

and 3.9, employs the Hessian at the MAP point for the spline coefficients to produce nonisotropic proposals. When the form of the prior covariance, mentioned above, and the standard likelihood expression for a Poisson process are used, the Hessian of the negative log posterior for $\boldsymbol{\beta}$ (given $r(t)$, τ_i and k) is given by

$$H_{ij} = a \sum_t B_i(t) B_j(t) + \sum_t B_i(t) \lambda(t) B_j(t), \tag{7.21}$$

where the first term is the inverse prior covariance and a is some positive constant. Since, by definition, $B_i(t)$ is nonzero only when $t \in [\tau_i, \tau_{i+4}]$, we see that H_{ij} vanishes when $|i - j| > 3$, and hence, is banded. Again, the bandedness of the Hessian is exploited (Wallstrom et al., 2007) to obtain the RWM proposals in $\mathcal{O}(T)$ computational time by the method described after equation 3.10. We note that the BARS package could potentially be improved by using a faster-mixing chain such as HMC, which can outperform RWM by orders of magnitude (see Figure 4).

We compared the mixing rates of the mentioned MCMC chains, in sampling from the posterior stimulus distributions for GLM-modeled neurons. In this setting, when the posterior is smooth throughout its support, the HMC algorithm outperforms the other chains by an order of magnitude. On the other hand, when sampling is from posteriors based on flat priors with sharp corners, the hit-and-run chain mixed consistently faster than the others.

In section 4, we compared the performance of the MAP and the posterior mean, calculated using MCMC, in different settings. In one example, we decoded simulated spike trains (generated in response to gaussian and flat white noise stimuli) in a range of stimulus input strengths and for different numbers of identical cells. We also decoded the filtered stimulus input into six retinal ganglion cells, based on their experimentally recorded spike trains. The average squared error of the MAP and mean estimates were in general quite close in the case of gaussian stimuli, justifying MAP decoding in this case. In the flat prior case, however, the posterior mean can often have a much smaller average squared error than the MAP.

In section 5, we applied MCMC to the problem of estimating properties of the joint distribution $p(\mathbf{x} \mid \boldsymbol{r})$ that cannot be obtained from its low-dimensional marginals. In particular, we investigated the reliability of the Laplace approximation for the mutual information between the stimulus and spike trains (model-based calculations of the mutual information with gaussian priors have been previously presented in Barbieri et al., 2004). We found that the Laplace approximation for the mutual information was adequate in the case of gaussian priors, except in a small range of moderate stimulus input strengths.

In the previous section we dealt with the effect of uncertainty in GLM parameters (e.g., based on fits to experimental data) on the decoding.

Intuitively, it is expected that when the forward model parameters become uncertain, information coming from the spike train, and hence the likelihood, becomes less reliable, and therefore the estimate will rely more heavily on the prior information. Thus, the posterior mean is expected to revert toward the prior mean as parameter uncertainty increases. We verified this intuition analytically in the special case of localized stimulus filters (with no bandpass filtering) and small baseline firing rates. Our numerics showed that the main systematic effect of increasing parameter uncertainty on the mean estimate, $E(\mathbf{x} \mid \boldsymbol{r})$, is to shrink its magnitude (thus sending to the origin, the prior mean in our case) in a wide range of parameter values.

The methods developed in this article and in Pillow et al. (2011), the companion article, can be used for a variety of applications. In future work we plan to further apply these techniques to other experimental data and to compare different "codebooks" (as mentioned in section 1) based on different reductions of the full spike trains according to their robustness and fidelity.

Acknowledgments

Thanks to M. Kennel, E. Simoncelli, E.J. Chichilnisky, T. Teravainen, and K. Rahnama Rad for helpful conversations. We are grateful to Y. Ding for her implementation of the bridge sampling method. Y.A. is supported by the Robert Leet and Clara Guthrie Patterson Trust Postdoctoral Fellowship, Bank of America, Trustee. L.P. is supported by NEI R01 EY018003, an Alfred P. Sloan Research Fellowship, and the McKnight Scholar award. J.P. is supported by a Royal Society USA/Canada Research Fellowship.

References

Abbott, L., & Dayan, P. (1999). The effect of correlated variability on the accuracy of a population code. *Neural Computation, 11*, 91–101.

Barbieri, R., Frank, L., Nguyen, D., Quirk, M., Solo, V., Wilson, M., et al. (2004). Dynamic analyses of information encoding in neural ensembles. *Neural Computation, 16*, 277–307.

Bennett, C. H. (1976). Efficient estimation of free energy divergences from Monte Carlo data. *Journal of Computational Physics, 22*, 245–268.

Berger, J. (1993). *Statistical decision theory and Bayesian analysis*. Berlin: Springer.

Bialek, W., Rieke, F., de Ruyter van Steveninck, R., & Warland, D. (1991). Reading a neural code. *Science, 252*, 1854–1857.

Boneh, A., & Golan, A. (1979). *Constraints' redundancy and feasible region boundedness by random feasible point generator (RFPG)*. Paper presented at the Third European Congress on Operations Research (EURO III). Amsterdam.

Brillinger, D. (1988). Maximum likelihood analysis of spike trains of interacting nerve cells. *Biological Cyberkinetics, 59*, 189–200.

Brillinger, D. (1992). Nerve cell spike train data analysis: A progression of technique. *Journal of the American Statistical Association, 87*, 260–271.

Brockwell, A., Rojas, A., & Kass, R. (2004). Recursive Bayesian decoding of motor cortical signals by particle filtering. *Journal of Neurophysiology, 91*, 1899–1907.

Brooks, S., & Gelman, A. (1998). General methods for monitoring convergence of iterative simulations. *J. Comput. Graph. Statist., 7*, 434–455.

Brown, E., Barbieri, R., Eden, U., & Frank, L. (2003). Likelihood methods for neural data analysis. In J. Feng (Ed.), *Computational neuroscience: A comprehensive approach* (pp. 253–286). London: CRC.

Brown, E., Frank, L., Tang, D., Quirk, M., & Wilson, M. (1998). A statistical paradigm for neural spike train decoding applied to position prediction from ensemble firing patterns of rat hippocampal place cells. *Journal of Neuroscience, 18*, 7411–7425.

Chan, K. S., & Ledolter, J. (1995). Monte Carlo EM estimation for time series models involving counts. *Journal of the American Statistical Association, 90*(429), 242–252.

Chichilnisky, E. (2001). A simple white noise analysis of neuronal light responses. *Network: Computation in Neural Systems, 12*, 199–213.

Cronin, B., Stevenson, I. H., Sur, M., & Kording, K. P. (2009). Hierarchical Bayesian modeling and Markov chain Monte Carlo sampling for tuning curve analysis. *J. Neurophysiol.*, 00379.2009.

Davis, R., & Rodriguez-Yam, G. (2005). Estimation for state-space models: An approximate likelihood approach. *Statistica Sinica, 15*, 381–406.

Dayan, P., & Abbott, L. (2001). *Theoretical neuroscience*. Cambridge, MA: MIT Press.

DiMatteo, I., Genovese, C., & Kass, R. (2001). Bayesian curve fitting with free-knot splines. *Biometrika, 88*, 1055–1073.

Donoghue, J. (2002). Connecting cortex to machines: Recent advances in brain interfaces. *Nature Neuroscience, 5*, 1085–1088.

Doucet, A., de Freitas, N., & Gordon, N. (Eds.). (2001). *Sequential Monte Carlo in practice*. Berlin: Springer.

Duane, S., Kennedy, A. D., Pendleton, B. J., & Roweth, D. (1987). Hybrid Monte Carlo. *Physics Letters B, 195*(2), 216–222.

Eden, U. T., Frank, L. M., Barbieri, R., Solo, V., & Brown, E. N. (2004). Dynamic analysis of neural encoding by point process adaptive filtering. *Neural Comput., 16*(5), 971–998.

Ergun, A., Barbieri, R., Eden, U., Wilson, M., & Brown, E. (2007). Construction of point process adaptive filter algorithms for neural systems using sequential Monte Carlo methods. *IEEE Transactions on Biomedical Engineering, 54*, 419–428.

Gamerman, D. (1997). Sampling from the posterior distribution in generalized linear mixed models. *Statistics and Computing, 7*(1), 57–68.

Gamerman, D. (1998). Markov chain Monte Carlo for dynamic generalised linear models. *Biometrika, 85*(1), 215–227.

Gelman, A. (2004). *Bayesian data analysis*. Boca Raton, FL: Chapman and Hall/CRC.

Geman, S., & Geman, D. (1984). Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 6*, 721–741.

Gerwinn, S., Macke, J. H., & Bethge, M. (2009). Bayesian population decoding of spiking neurons. *Front. Comput. Neurosci., 3*(21), 1–28.

Geyer, C. J. (1992). Practical Markov chain Monte Carlo. *Statistical Science, 7*, 473–483.

Gilks, W. R. (1992). Derivative-free adaptive rejection sampling for Gibbs sampling. In J. Bernardo, J. Berger, A. Dawid, & A. Smith (Eds.), *Bayesian statistics 4* (pp. 641–649). New York: Oxford University Press.

Gilks, W., & Wild, P. (1992). Adaptive rejection sampling for Gibbs sampling. *Applied Statistics, 41*, 337–348.

Godsill, S., Doucet, A., & West, M. (2004). Monte Carlo smoothing for non-linear time series. *Journal of the American Statistical Association, 99*, 156–168.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika, 57*, 97–109.

Huys, Q., & Paninski, L. (2009). Smoothing of, and parameter estimation from, noisy biophysical recordings. *PLoS Comput. Biol., 5*(5), e1000379.

Ishwaran, H. (1999). Applications of hybrid Monte Carlo to Bayesian generalized linear models: Quasicomplete separation and neural networks. *Journal of Computational and Graphical Statistics, 8*, 779–799.

Jacobs, A., Grzywacz, N., & Nirenberg, S. (2006). Decoding the parallel pathways of the retina. *Society for Science Abstracts*.

Jungbacker, B., & Koopman, S. J. (2007). Monte Carlo estimation for nonlinear non-gaussian state space models. *Biometrika, 94*(4), 827–839.

Karmeier, K., Krapp, H., & Egelhaaf, M. (2005). Population coding of self-motion: Applying Bayesian analysis to a population of visual interneurons in the fly. *Journal of Neurophysiology, 94*, 2182–2194.

Kass, R., Tierney, L., & Raftery, A. (1991). Laplace's method in Bayesian analysis. In N. Flournoy & R. Tsutakawa (Eds.), *Statistical multiple integration* (pp. 89–99). Providence, RI: American Mathematical Society.

Kelly, R., & Lee, T. (2004). Decoding V1 neuronal activity using particle filtering with Volterra kernels. In S. Becker, S. Thrün, & K. Obermayer (Eds.), *Advances in neural information processing systems, 15*, 1359–1366.

Kennedy, A., Edwards, R., Mino, H., & Pendleton, B. (1996). Tuning the generalized hybrid monte carlo algorithm. *Nuclear Physics B–Proceedings Supplements, 47*(1–3), 781–784.

Kipnis, C., & Varadhan, S. R. S. (1986). Central limit theorem for additive functionals of reversible Markov processes and applications to simple exclusions. *Comm. Math. Phys., 104*, 1–19.

Litke, A., Bezayiff, N., Chichilnisky, E., Cunningham, W., Dabrowski, W., Grillo, A., et al. (2004). What does the eye tell the brain? Development of a system for the large scale recording of retinal output activity. *IEEE Trans. Nucl. Sci., 2*, 1434–1440.

Lovasz, L., & Vempala, S. (2004). Hit-and-run from a corner. In *Proc. of the 36th ACM Symposium on the Theory of Computing (STOC '04)*. New York: ACM.

Maynard, E., Hatsopoulos, N., Ojakangas, C., Acuna, B., Sanes, J., Normann, R., et al. (1999). Neuronal interactions improve cortical population coding of movement direction. *Journal of Neuroscience, 19*, 8083–8093.

McCullagh, P., & Nelder, J. (1989). *Generalized linear models*. London: Chapman and Hall.

Meng, X. L., & Wong, W. H. (1996). Simulating ratios of normalizing constants via a simple identity: A theoretical exploration. *Statistica Sinica, 6*, 831–860.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of state calculations by fast computing machines. *J. Chem. Phys., 21*, 1087–1092.

Neal, R. (1996). *Bayesian learning for neural networks*. Berlin: Springer.

Neal, R. (2008). The harmonic mean of the likelihood: Worst Monte Carlo method ever. *Radford Neal's blog*. Available online at http://radfordneal.wordpress.com/2008/08/17/the-harmonic-mean-of-the-likelihood-worst-monte-carlo-method-ever/.

Newton, M. A., & Raftery, A. E. (1994). Approximate Bayesian inference with the weighted likelihood bootstrap. *Journal of the Royal Statistical Society. Series B (Methodological), 56*(1), 3–48.

Paninski, L. (2003). Estimation of entropy and mutual information. *Neural Computation, 15*, 1191–1253.

Paninski, L. (2004). Maximum likelihood estimation of cascade point-process neural encoding models. *Network: Computation in Neural Systems, 15*, 243–262.

Paninski, L., Ahmadian, Y., Ferreira, D., Koyama, S., Rahnama Rad, K., Vidne, M., Vogelstein, J., & Wu, W. (2010). A new look at state-space models for neural data. *Journal of Computational Neuroscience, 29*(1), 107–126.

Paninski, L., Fellows, M., Shoham, S., Hatsopoulos, N., & Donoghue, J. (2004). Superlinear population encoding of dynamic hand trajectory in primary motor cortex. *J. Neurosci., 24*, 8551–8561.

Paninski, L., Iyengar, S., Kass, R., & Brown, E. (2008). Statistical models of spike trains. In C. Laing & G. Lord (Eds.), *Stochastic methods in neuroscience*. New York: Oxford University Press.

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

Pillow, J., Shlens, J., Paninski, L., Sher, A., Litke, A., Chichilnisky, E., et al. (2008). Spatiotemporal correlations and visual signaling in a complete neuronal population. *Nature, 454*, 995–999.

Reid, R. C., Victor, J. D., & Shapley, R. M. (1997). The use of m-sequences in the analysis of visual neurons: Linear receptive field properties. *Visual Neuroscience, 14*(6), 1015–1027.

Rieke, F., Warland, D., de Ruyter van Steveninck, R., & Bialek, W. (1997). *Spikes: Exploring the neural code*. Cambridge, MA: MIT Press.

Robert, C., & Casella, G. (2005). *Monte Carlo statistical methods*. Berlin: Springer.

Roberts, G., & Rosenthal, J. (1998). Optimal scaling of discrete approximations to Langevin diffusions. *J. R. Statist. Soc. B, 160*, 255–268.

Roberts, G., & Rosenthal, J. (2001). Optimal scaling for various Metropolis-Hastings algorithms. *Statistical Science, 16*, 351–367.

Roberts, G. O., & Tweedie, R. L. (1996). Exponential convergence of Langevin diffusions and their discrete approximations. *Biometrika, 2*, 341–363.

Sanger, T. (1994). Theoretical considerations for the analysis of population coding in motor cortex. *Neural Computation, 6*, 12–21.

Shephard, N., & Pitt, M. K. (1997). Likelihood analysis of non-gaussian measurement time series. *Biometrika, 84*(3), 653–667.

Shlens, J., Field, G. D., Gauthier, J. L., Grivich, M. I., Petrusca, D., Sher, A., et al. (2006). The structure of multi-neuron firing patterns in primate retina. *J. Neurosci., 26*(32), 8254–8266.

Shoham, S., Paninski, L., Fellows, M., Hatsopoulos, N., Donoghue, J., & Normann, R. (2005). Optimal decoding for a primary motor cortical brain-computer interface. *IEEE Transactions on Biomedical Engineering, 52*, 1312–1322.

Simoncelli, E., Paninski, L., Pillow, J., & Schwartz, O. (2004). Characterization of neural responses with stochastic stimuli. In M. S. Gazzaniga (Ed.), *The cognitive neurosciences* (3rd ed.). Cambridge, MA: MIT Press.

Smith, A. C., Frank, L. M., Wirth, S., Yanike, M., Hu, D., Kubota, Y., et al. (2004). Dynamic analysis of learning in behavioral experiments. *J. Neurosci., 24*(2), 447–461.

Smith, A. C., Wirth, S., Suzuki, W. A., & Brown, E. N. (2007). Bayesian analysis of interleaved learning and response bias in behavioral experiments. *J. Neurophysiol., 97*(3), 2516–2524.

Smith, R. L. (1980). *Monte Carlo techniques for generating random feasible solutions to mathematical programs*. Paper presented at the ORSA/TIMS conference, Washington, DC.

Snyder, D., & Miller, M. (1991). *Random point processes in time and space*. Berlin: Springer-Verlag.

Stanley, G., & Boloori, A. (2001). Decoding in neural systems: stimulus reconstruction from nonlinear encoding. In *Proceedings of the Annual EMBS International Conference* (p. 23). Piscataway, NJ: EMBS.

Strong, S. Koberle, R., de Ruyter van Steveninck R., & Bialek, W. (1998). Entropy and information in neural spike trains. *Physical Review Letters, 80*, 197–202.

Theunissen, F., David, S., Singh, N., Hsu, A., Vinje, W., & Gallant, J. (2001). Estimating spatio-temporal receptive fields of auditory and visual neurons from their responses to natural stimuli. *Network: Computation in Neural Systems, 12*, 289–316.

Tierney, L. (1991). *Markov chains for exploring posterior distributions* (Tech. Rep. 560). Minneapolis: School of Statistics, University of Minnesota.

Tierney, L., & Kadane, J. B. (1986). Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association, 81*(393), 82–86.

Truccolo, W., Eden, U., Fellows, M., Donoghue, J., & 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.

Tyler, D. E. (1987). Statistical analysis for the angular central gaussian distribution on the sphere. *Biometrika, 74*, 579–859.

Vogelstein, J., Packer, A., Machado, T., Sippy, T., Babadi, B., Yuste, R., et al. (in press). Fast nonnegative deconvolution for spike train reference from calcium imaging. *J. Neurophys.*

Wallstrom, G., Liebner, J., & Kass, R. E. (2007). An implementation of Bayesian adaptive regression splines (BARS) in C with S and R wrappers. *Journal of Statistical Software, 26*(1), 1–21.

Warland, D., Reinagel, P., & Meister, M. (1997). Decoding visual information from a population of retinal ganglion cells. *Journal of Neurophysiology, 78*, 2336–2350.

Wu, W., Black, M. J., Mumford, D., Gao, Y., Bienenstock, E., & Donoghue, J. (2004). Modeling and decoding motor cortical activity using a switching Kalman filter. *IEEE Transactions on Biomedical Engineering, 51*, 933–942.

Yu, B. M., Cunningham, J. P., Santhanam, G., Ryu, S. I., Shenoy, K. V., & Sahani, M. (2009). Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. *J. Neurophysiol., 102*(1), 614–635.

Zhang, K., Ginzburg, I., McNaughton, B., & Sejnowski, T. (1998). Interpreting neuronal population activity by reconstruction: Unified framework with application to hippocampal place cells. *Journal of Neurophysiology, 79*, 1017–1044.

Received December 24, 2007; accepted May 3, 2010.