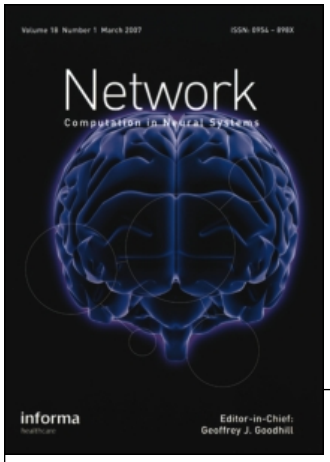

Network: Computation in Neural Systems
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713663148

Common-input models for multiple neural spike-train data

Jayant E. Kulkarni [a]; Liam Paninski [ab]
[a] Center for Theoretical Neuroscience, Columbia University, New York, USA
[b] Department of Statistics, Columbia University, New York, USA

Online Publication Date: 01 January 2007

To cite this Article: Kulkarni, Jayant E. and Paninski, Liam (2007) 'Common-input models for multiple neural spike-train data', Network: Computation in Neural Systems, 18:4, 375 — 407

To link to this article: DOI: 10.1080/09548980701625173
URL: http://dx.doi.org/10.1080/09548980701625173

Network: Computation in Neural Systems
December 2007; 18(4): 375–407

informa
healthcare

Common-input models for multiple neural spike-train data

JAYANT E. KULKARNI[1] & LIAM PANINSKI[1,2]

[1]*Center for Theoretical Neuroscience, Columbia University, New York, USA and*
[2]*Department of Statistics, Columbia University, New York, USA*

(Received 30 July 2007; accepted 13 August 2007)

Abstract
Recent developments in multi-electrode recordings enable the simultaneous measurement of the spiking activity of many neurons. Analysis of such multineuronal data is one of the key challenge in computational neuroscience today. In this work, we develop a multivariate point-process model in which the observed activity of a network of neurons depends on three terms: (1) the experimentally-controlled stimulus; (2) the spiking history of the observed neurons; and (3) a hidden term that corresponds, for example, to *common input* from an unobserved population of neurons that is presynaptic to two or more cells in the observed population. We consider two models for the network firing-rates, one of which is computationally and analytically tractable but can lead to unrealistically high firing-rates, while the other with reasonable firing-rates imposes a greater computational burden. We develop an expectation-maximization algorithm for fitting the parameters of both the models. For the analytically tractable model the expectation step is based on a continuous-time implementation of the extended Kalman smoother, and the maximization step involves two concave maximization problems which may be solved in parallel. The other model that we consider necessitates the use of Monte Carlo methods for the expectation as well as maximization step. We discuss the trade-off involved in choosing between the two models and the associated methods. The techniques developed allow us to solve a variety of inference problems in a straightforward, computationally efficient fashion; for example, we may use the model to predict network activity given an arbitrary stimulus, infer a neuron's ring rate given the stimulus and the activity of the other observed neurons, and perform optimal stimulus decoding and prediction. We present several detailed simulation studies which explore the strengths and limitations of our approach.

Keywords: *Common-input, network model, cox process, expectation-maximization algorithm*

Correspondence: Jayant E. Kulkarni, Center for Theoretical Neuroscience, Columbia University, 1051, Riverside Drive, Unit 87, New York, 10032 USA. E-mail: jk2619@columbia.edu

ISSN 0954-898X print/ISSN 1361-6536 online/07/040375–407 © 2007 Informa UK Ltd.
DOI: 10.1080/09548980701625173

Introduction

With the advent of large-scale multi-electrode recordings, it is now possible to collect simultaneous spiking data from large ensembles of neurons. This multineuronal spike-train data gives us the ability to investigate important questions facing systems neuroscience. For example, we can now address in detail the question of how networks of neurons process complex dynamic inputs and encode information (Abeles 1991; Schnitzer and Meister 2003; Cossart et al. 2003; Litke et al. 2004). Understanding the concerted activity of large networks will also help in the design of neural prosthetic devices which have significant clinical implications (Loizou 1998; Donoghue 2002; Weiland et al. 2005).

To achieve these goals we need to develop tractable, powerful methods for modeling multineuronal spike-train data (Brown et al. 2004). There is a large body of literature tackling the problem of developing statistical models for large-scale simultaneous spike-train recordings (Chornoboy et al. 1988; Utikal 1997; Martignon et al. 2000; Iyengar 2001; Schnitzer and Meister 2003; Nicolelis et al. 2003; Cossart et al. 2003; Paninski et al. 2004; Truccolo et al. 2005; Okatan et al. 2005; Pillow et al. 2005; Nykamp 2005). Nearly all of these previous efforts at developing population spike-train models (with the notable exception of Nykamp (2005)) have taken the basic form

$$\lambda_k(t) = F_k(\vec{x}(t), \vec{n}(t)),$$

where $\lambda_k(t)$ represents the instantaneous firing rate of the k-th observed neuron, and $F_k(\cdot)$ is some function that relates $\lambda_k(t)$ to the simultaneously observed *external* $\vec{x}(t)$ and *internal* $\vec{n}(t)$ signals. Typically, $\vec{x}(t)$ includes a truncated history of the presented stimulus along with measurements of the animal's behavioral state, while $\vec{n}(t)$ could include the firing rates of all the other observed neurons, and/or measurements of multiunit activity or local field potential (Andersen et al. 2004).

Thus, most of the models have stimulus-dependence terms and *direct-coupling* terms representing the influence that the activity of an observed cell might have on the other recorded neurons. Fewer models, however, have attempted to include the effects of the population of neurons which are not directly observed during the experiment (Nykamp 2005). Since we can directly observe only a small fraction of neurons in any physiological preparation, such unmeasured neurons might have a large collective impact on the dynamics and coding properties of the observed neural population. For example, it is well-understood that *common input* effects play an essential role in the interpretation of pairwise cross-correlograms (Brody 1999; Dayan and Abbott 2001; Nykamp 2005), and that these effects can be expected to become even more important with an increase in the size of the observed neural population.

Here, we consider models in which the firing rates of the neurons depend not only on the stimulus history and the spiking history of the observed neurons but also on common inputs. The models are a multivariate version of a Cox process, also known as a doubly-stochastic point process (Cox 1955; Snyder and Miller 1991; Moeller et al. 1998; Moeller and Waagepetersen 2004). Related models have seen several applications in the fields of neural information processing and neural data analysis (Smith and Brown 2003; Jackson 2004; Brockwell et al. 2004; Zemel et al. 2004;

Sahani 1999; Wu et al. 2004, 2005; Yu et al. 2006). To our knowledge, however, our work represents the first application of these techniques to the common-input population model.

In this article we consider two models which differ in the way they map the *internal* and *external* signals to the firing-rates of the neurons. The two models considered elucidate a key trade-off between computational and analytical tractability, and biophysical fidelity. The tractable model can lead to unrealistic firing-rates, while the model with reasonable firing-rates is computationally expensive. In this article we derive the methodology required to consider both approaches. While we present a few simulation results for Model 2, the thrust of this article is to develop computationally tractable models, and we focus primarily on Model 1 in the results section.

To fit the tractable model we derive a modification of the standard expectation–maximization (EM) algorithm (Dempster et al. 1977; Smith and Brown 2003) paying special attention to computational efficiency. The EM algorithm alternates between a run of the the expectation step (E-step) and the maximization step (M-step). The E-step computes the observation-conditioned distribution of the hidden process. It is implemented using a forward-sweep which computes the conditional distributions at time t using observations up to time t, and a backward-sweep which computes the fully-conditioned distribution. The M-step comprises of two optimization problems, both of the optimization problems have unique solutions. One of the two problems can be solved analytically. The other reduces into K smaller, independent, strictly-convex optimization problems (K denotes the number of observed neurons) which can all be solved in parallel.

In the case of the model with greater biophysical fidelity, the backward E-step remains the same, but the forward sweep of the E-step is carried out using Monte Carlo techniques with importance sampling. The M-step which involves the computation of certain expectations is also carried out using Monte Carlo techniques. This stochastic approach make parameter estimation for this model more computationally expensive.

Once the parameters have been obtained we may use the models to solve a variety of important stimulus-response inference problems: for example, we may use the model to predict activity in the network (including firing rates, cross-correlations, etc.) given some arbitrary stimulus $x(t)$, or alternatively perform optimal stimulus decoding the given observed network activity (Pillow and Paninski 2005; Truccolo et al. 2005).

This article is structured as follows: Section ''Theory'' develops the theory used to estimate the parameters for our models. In this section we present the background for the Cox process model as well the EM algorithm. The Fokker–Planck approach for numerically exact computation of the forward distributions is presented briefly. This exact but computationally expensive approach is abandoned in favor of the extended Kalman smoother (EKS) which is dealt with in more detail. The optimization problems involved in the M-step are also presented. In section ''Simula results'' we present results from detailed simulation studies which explore the strengths and limitations of the proposed approach. We consider the two models and study the accuracy of the estimates using the methods developed for these two models. We close in section

"Discussion" by pointing out various connections with previous work and discussing several important directions for future research.

Theory

In this section we outline the mathematical structure of the model and summarize the techniques used to estimate the model parameters.

The multivariate Cox process model

We consider a multivariate point process model (Snyder and Miller 1991) for a network of neurons (Figure 1) whose conditional intensity function is given by $\vec{\lambda}(t) = f(\vec{V}(t))$, where $\vec{V}(t) = \vec{I}(t) + G\vec{N}(t)$. In component form, the conditional intensity function for a neuron k at time t is given by

$$\lambda_k(t) = f(V_k(t)), \tag{1}$$

where $V_k(t) = (I_k(t) + \vec{g}_k\vec{N}(t))$; $\vec{g}_k$ denotes the k-th row of the matrix G; f is a fixed, smoothly rectifying nonlinearity; and the terms $\vec{I}(t)$ and $\vec{N}(t)$ correspond to the fully-observed and unobserved parts of the model, respectively. The log-likelihood for such point-processes is given by

$$L(\{t_{k,j}\}) = \log p(\{t_{k,j}\}|\{\lambda_k(t)\}) = \sum_{k=1}^{K}\left(\sum_j \log \lambda_k(t_{k,j}) - \int_0^T \lambda_k(s)\mathrm{d}s\right) + \text{const.}, \tag{2}$$

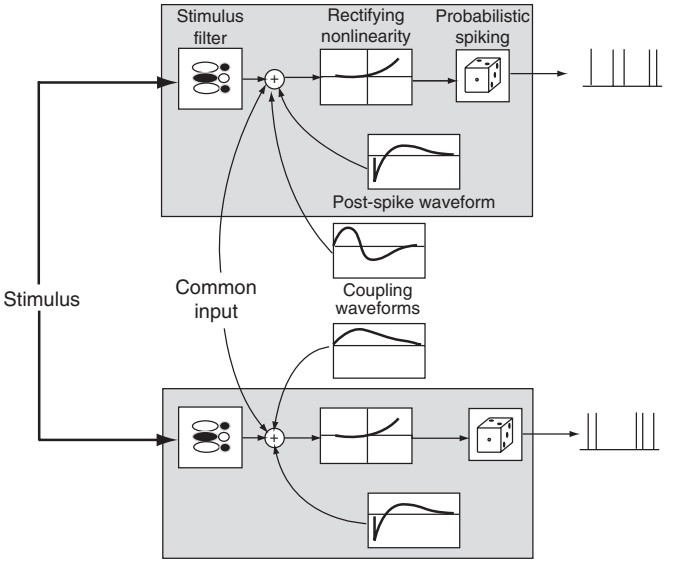

Figure 1. Generalized Linear Model (GLM): The input to a neuron is the sum of the linearly-filtered experimentally-controlled stimulus, the interneuronal cross-coupling terms, the neuron's post-spike waveform, and unobserved common-input terms. The post-spike waveform can account for effects such as refractoriness and adaptation, while the common-input terms can capture input from a shared pool of unobserved neurons presynaptic to the observed neurons. This summated input to each neuron in the observed population is passed through a point nonlinearity which drives the probabilistic spike-generation point process.

where $t_{k,j}$ is the time of the j-th spike observed from the k-th neuron and $[0, T)$ is the time interval over which we observe the spiking responses.

The observed component of the model, $I_k(t)$, captures the dependence of the firing rate on the externally applied stimulus and cross-coupling terms from other neurons in the observed population. As described in previous work (Brillinger 1988; Paninski 2004; Truccolo et al. 2005), we let $I_k(t)$ take the form,

$$I_k(t) = \vec{r}_k \cdot \vec{x}(t) + \sum_{j=1}^{K} \sum_{\{i\,:\,t_{j,i}<t\}} h_{kj}(t - t_{j,i}), \tag{3}$$

where $\vec{x}(t)$ is the stimulus at time t, $\vec{r}_k$ is the k-th cell's receptive field, and $h_{kj}(\cdot)$ denotes the interneuronal coupling term when $k \neq j$ and cell k's post-spike current term for $k = j$. Thus, the fully-observable input to each neuron is a sum of the linearly-filtered stimulus-dependent signal and the network's spiking history; the parameters $\{\vec{r}_k\}$ and $\{h_{kj}(.)\}$ act as linear weights here. Incorporating terms corresponding to the neuron's own spike history allows us to model various intrinsic spiking effects; such as refractoriness, burstiness, and adaptation; while incorporating interneuronal *cross-terms* can capture more intricate reverberatory and/or locally inhibitory network effects.

The unobserved signal $\vec{N}(t)$, on the other hand, is a vector that represents the lumped activity of a population of presynaptic neurons. The matrix G is a set of weights that couples the dynamics of the hidden common-input process $\vec{N}(t)$ to the instantaneous firing rate $\vec{\lambda}(t)$. If we pretend for the moment that $\vec{N}(t)$ is fully-observed, then the model reduces to a standard generalized linear model (GLM), as described in detail and considered by many previous authors (Chornoboy et al. 1988; Utikal 1997; Paninski et al. 2004; Truccolo et al. 2005; Okatan et al. 2005). As discussed in Paninski (2004), this model can be fit by maximizing the log-likelihood (2) by solving a single multivariate convex optimization problem whenever the link function $f(\cdot)$ is chosen, so that $f(\cdot)$ is convex and $\log f(\cdot)$ is concave (e.g., $f(u) = \exp(u)$). Thus, fitting this fully-observed model is highly tractable. Moreover, previous authors have discussed this GLM in the context of a simplified "soft-threshold" approximation to the canonical integrate-and-fire model (Plesser and Gerstner 2000; Stevens and Zador 1996; Paninski 2004; Paninski et al. 2008).

However, in the present case, $\vec{N}(t)$ is not observed, so we must hypothesize some tractable, flexible model for these common-input terms. Here, for computational tractability, we assume that the hidden process can be defined by Gauss–Markov dynamics,

$$\frac{\mathrm{d}\vec{N}(t)}{\mathrm{d}t} = -D\vec{N}(t) + \vec{Z}(t), \tag{4}$$

where the input process $\{\vec{Z}(t), t \geq 0\}$ is $\mathcal{J}$-dimensional standard Gaussian white-noise, and D is a matrix, which, for stability reasons, has eigenvalues whose real parts are nonpositive. This linear model may appear to be an oversimplification but if the dimension $\mathcal{J}$ of $\vec{N}(t)$ is sufficiently large then this model can lead to a rich repertoire of dynamic behavior, including oscillatory behavior, with correlations on multiple time-scales. The Gaussian assumption is justified by the central limit theorem, as we assume that the hidden input to any single neuron is the weighted sum of small inputs from a large population.

The two models that we consider differ in the link function, that is the rectifying non-linearity $f(\cdot)$, which maps the external and internal signals to the firing-rates. The choice of the link function is crucial as it predominantly sets the statistics of the spiking process. Different choices of this function will entail different approaches in determining the observation-conditioned densities of the hidden process which are needed in the M-step of the EM algorithm. The two models that we consider are:

$$\text{Model 1:} \quad \lambda_k(t) = \exp(V_k(t)) \tag{5}$$

$$\text{Model 2:} \quad \lambda_k(t) = \begin{cases} \exp(V_k(t)) & \text{if } V_k(t) < 0 \\ \left(1 + V_k(t) + V_k(t)^2/2\right) & \text{if } V_k(t) \geq 0 \end{cases} \tag{6}$$

where $V_k = (I_k + \vec{g}_k \vec{N})$ as described earlier. The only difference between the two models is that in Model 2 we retain a second-order approximation to the exponential nonlinearity when $V_k \geq 0$. Model 1 with its exponential non-linearity can lead to unrealistically firing rates. This biophysical implausibility is mitigated in Model 2 due to the truncation of the exponential to the second order. While the difference in the two models is conceptually minor, the approaches used in determining the conditional distributions of the hidden process, are significantly different. As we will see, Model 1 allows us to compute most of the terms in the forward-filter analytically, and those that cannot be computed analytically can be approximated readily. For Model 2, on the other hand, we have to resort to computationally expensive Monte Carlo methods. The consideration of Model 2 elucidates how the methods introduced in this article can be implemented for more general functions by the introduction of Monte Carlo methods.

The spike-generation model is a doubly-stochastic process: spikes are generated stochastically and the process $\vec{N}(t)$ driving the spike-generation is also stochastic. Such doubly-stochastic point processes are also known as Cox processes (Snyder and Miller 1991; Moeller and Waagepetersen 2004). Because of our assumption (4) of Markovian dynamics for the hidden variable $\vec{N}(t)$, our model is in fact a type of hidden Markov process, related more specifically, to the Kalman filter model. Kalman filter theory for state-space models with point process observations has been well developed (Snyder and Miller 1991; Smith and Brown 2003) and can be fruitfully applied to our model. Though all the parameters affect the spiking activity of the network, the parameters $\{\vec{r}_k\}, \{h_{k,j}\}$ and G have a direct effect while the effect of the dynamics matrix D is indirect as it modulates the firing rate only through the latent process. Keeping this in mind, for convenience, in the text we will refer to $\{\vec{r}_k\}, \{h_{k,j}\}$ and G as the intensity-parameters and D as the dynamic-parameter.

Before we proceed with the EM algorithm, we must first restrict our model somewhat, as the model parameters $\theta = \{D, G, \{\vec{r}_k\}, \{h_{kj}\}\}$ are known to be nonidentifiable (Roweis and Ghahramani 1999). That is, two different settings θ_1, θ_2 of the model parameters may give rise to exactly the same data distribution, $p(\{t_{k,j}\}|\{\vec{x}(t)\}, \theta_1) = p(\{t_{k,j}\}|\{\vec{x}(t)\}, \theta_2)$. Specifically, the model currently has the following symmetry: if we redefine $\vec{N}(t)$ by an orthogonal change of basis, $\vec{N} \to O\vec{N}$, for some orthogonal matrix O, then we may simply redefine $G \to GO'$ and not see any difference in the observed data, since the observed data depend only on $G\vec{N}$, which is clearly preserved by the transformation $(\vec{N}, G) \to (O\vec{N}, GO')$, since

$GO'O\vec{N} = G\vec{N}$. Thus, our model is only defined uniquely up to an orthogonal change of basis O. To remove these extra unidentifiable degrees of freedom in θ we simply restrict G to be a lower-triangular matrix, with the diagonal elements restricted to be non-negative; this simple (convex) constraint on G restores the identifiability of the model. With this lower-triangular constraint on G, and D constrained to have eigenvalues whose real parts are non-negative, while the other parameters $\{\{\vec{r}_k\}, \{h_{kj}\}\}$ are unconstrained, we have a convex set of parameters to search over. We note here that other parametrizations are also possible, for example we could fix G and vary the covariance of $Z(t)$.

Expectation–maximization (EM) algorithm

As emphasized above, fitting this model by maximizing the loglikelihood (2) is quite straightforward when $\vec{N}(t)$ is fully observed. However, in our case, $\vec{N}(t)$ is not observed, and we must instead maximize the marginal likelihood

$$p(\{t_{k,j}\}|\theta) = \int p(\{t_{k,j}\}|\vec{N}_{0,T}, \theta) p(\vec{N}_{0,T}|\theta) \mathrm{d}\vec{N}_{0,T}, \tag{7}$$

where $\vec{N}_{0,T}$ denotes the hidden sequence $\vec{N}_{0,T} = \{\vec{N}(t)\}_{0 \le t \le T}$. We develop an EM algorithm (Dempster et al. 1977; Smith and Brown 2003) to estimate the parameters θ. A variant of this algorithm (Salakhutdinov et al. 2003) may also be used to compute the gradients of the marginal likelihood (7), in cases where it is useful to maximize this likelihood directly by gradient ascent methods.

We begin by defining some notation. Let the observed spiking activity of neuron k, in a network with K neurons, over the interval $[0, T)$ be denoted by $O^k_{0,T}$; a right-continuous counting function with discontinuities at the spike-times of the neuron (Snyder and Miller 1991). The ensemble spiking-activity is given by $O_{0,T} = \{O^1_{0,T}, O^2_{0,T} \ldots, O^K_{0,T}\}$. Then our marginal loglikelihood may be written as

$$\begin{aligned} \log p(\{t_{k,j}\}|\theta) &= \log[p(O_{0,T}|\theta)] \\ &= \log\left[\int p(O_{0,T}, \vec{N}_{0,T}|\theta) \mathrm{d}\vec{N}_{0,T}\right]. \end{aligned} \tag{8}$$

The EM algorithm ascends this log-likelihood by the construction of an auxiliary function $Q(\theta|\theta_{i-1})$ given by

$$\begin{aligned} Q(\theta|\theta_{i-1}) &= \mathrm{E}\left[\log[p(O_{0,T}, \vec{N}_{0,T}|\theta)] \middle| O_{0,T}, \theta_{i-1}\right] \\ &= \mathrm{E}\left[\log\left[p(\vec{N}(0)|\theta) \prod_{t=\delta t}^{T} \left(p(\vec{N}(t)|\vec{N}(t-\delta t), \theta) p(O_t|\vec{N}(t), \theta)\right)\right] \middle| O_{0,T}, \theta_{i-1}\right] \\ &= \mathrm{E}\left[\log\left[p(\vec{N}(0)|\theta)\right] + \sum_{t=\delta t}^{T} \Big(\log p(\vec{N}(t)|\vec{N}(t-\delta t), \theta) \right. \\ &\quad \left. + \log p(O_t|\vec{N}(t), \theta)\Big) \middle| O_{0,T}, \theta_{i-1}\right], \end{aligned} \tag{9}$$

where δt is a small time-step which we are free to choose. Later on we discuss the use of adaptive time-step methods (section "Forward E-step: Kalman filter") to minimize computational costs.

The EM algorithm is an iterative scheme involving alternate runs of the E-step and M-step given initial estimates for the parameters θ_0 (Dempster et al. 1977; Smith and Brown 2003). For the Kalman filter (and hidden Markov models more generally) the EM algorithm has been described extensively in a number of different contexts (Snyder 1972a, b, Rabiner 1989; Harvey 1991; Doucet et al. 2001; Smith and Brown 2003; Yu et al. 2006; Wu et al. 2005). The E-step involves the computation of the fully-observed distributions given the parameter estimates θ_{i-1} from the previous iteration, while the M-step involves maximizing $Q(\theta|\theta_{i-1})$ over θ (with θ_{i-1} held fixed) to obtain parameter estimates θ_i for the next iteration step.

As is well-known, the E-step may be broken into two stages, known as the *forward* and *backward* step. In the *forward* step we compute the forward density $p(\vec{N}(t)|O_{0,t}, \theta_{i-1})$ recursively for $t \in [0, T]$, beginning with some initial conditions $p(\vec{N}(0)|\theta_{i-1})$. The *backward* step recursively modifies the forward density to compute the fully-conditioned probabilities $p(\vec{N}(t))|O_{0,T}, \theta_{i-1})$ and pairwise probabilities $p(\vec{N}(t), \vec{N}(t+\delta t)|O_{0,T}, \theta_{i-1})$ for all $t \in [0, T]$, starting with the end conditions $p(\vec{N}(T)|O_{0,T}, \theta_{i-1})$ and recursing backward from $t = T$ to $t = 0$. Note that the end-condition for the backward-step $p(\vec{N}(T)|O_{0,T}, \theta_{i-1})$ is obtained from the forward-step. Due to the Markov nature of the Model (9), these fully-conditioned probabilities are all we need to compute the expectations in $Q(\theta|\theta_{i-1})$.

For models such as the one considered here, where the hidden process is linear Gaussian, the backward step is fairly standard, and essentially *smoothes* the densities obtained via the forward-sweep using the dynamics of the hidden process. The backward densities may be computed via simple manipulations of mean vectors and covariance matrices of certain Gaussian distributions (Mendel 1995; de Jong and MacKinnon 1988; Smith and Brown 2003).

The forward step, on the other hand, is slightly more subtle. In the standard Kalman filter, with linear Gaussian observations, the forward density is Gaussian for all times $t \in [0, T]$. Thus we do not need to keep track of the full shape of this density for all t; instead, we only need to track the mean and covariance of this Gaussian. It is also straightforward to compute these quantities in a recursive manner. In our case, unfortunately, the non-Gaussianity of the observations implies that the forward density is also non-Gaussian, and to track this density exactly we need to keep track of more than just the mean and the covariance. Alternately, we may take an approximate approach: we approximate this density as a Gaussian, and at each time step just track the mean and covariance. Each approach has its strengths and weaknesses. In addition, there are many different ways to construct this Gaussian approximation, as we will discuss below. In the following two subsections, we describe: (1) a method for numerically computing the forward density exactly, and (2) a method for recursively computing the Gaussian approximation.

The Expectation step

As discussed earlier, the EM algorithm requires the knowledge of the fully-conditioned distributions $p(\vec{N}(t)|O_{0,T}, \theta_{i-1})$ and $p(\vec{N}(t), \vec{N}(t+\delta t)|O_{0,T}, \theta_{i-1})$ at

each instant in time. There are different ways in which these distributions can be computed, and the classical approach involves a forward-sweep over the data followed by a backward-sweep. In this section we discuss how these forward and backward steps are implemented along with the computationally more intensive method of numerically solving the Fokker–Planck equation for the forward step.

Forward E-step: Fokker–Planck approach. An application of the Feynman–Kac formula (Karatzas and Shreve 1997) shows that the joint density $P(\vec{N}, t) \equiv p(\vec{N}(t), O_{0,t})$ may be computed, up to an irrelevant constant factor, via the following Fokker–Planck equation,

$$\frac{\partial P(\vec{N}, t)}{\partial t} = \frac{\mathrm{div} P(\vec{N}, t)}{2} - \sum_{j=1}^{\mathcal{J}} \frac{\partial [D\vec{N} P(\vec{N}, t)]}{\partial N^j} - \sum_{k=1}^{K} f[I_k(t) + \vec{g}_k \vec{N}] P(\vec{N}, t), \tag{10}$$

where N^j is the j-th element of $\vec{N}$, with the time-discontinuous update rule

$$P(\vec{N}, t_{k,j}^+) = P(\vec{N}, t_{k,j}^-) f[I_k(t_{k,j}) + \vec{g}_k \vec{N}] \tag{11}$$

at time $t_{k,j}$, for the j-th spike time observed from the k-th cell. This PDE has boundary condition

$$\int P(\vec{N}, t) \mathrm{d}\vec{N} < \infty. \tag{12}$$

In the absence of any spiking observations a reasonable initialization $P(\vec{N}, 0)$ may be constructed by using the asymptotic mean and covariance of the latent process $\vec{N}(t)$. The solution $\{\vec{N}(t), t \geq 0\}$ to Equation 4 is a Gaussian process with moments

$$\mathrm{E}[\vec{N}(t)] = \exp(-tD)\vec{N}(0), \tag{13}$$

$$\mathrm{Cov}[\vec{N}(t)] = \frac{1}{2} D^{-1} (I - \exp(-2tD)), \tag{14}$$

where I is the appropriately-sized identity matrix. Therefore, we may take $P(\vec{N}, 0)$ to be Gaussian with mean 0 and covariance $\frac{1}{2} D^{-1}$.

While mathematically elegant, solving this equation when $\dim(\vec{N}) > 2$ becomes computationally very expensive. Thus we will describe a more efficient, albeit approximate, technique known as the extended Kalman smoother (EKS) for computing $P(\vec{N}, t)$ in the next section. For $\dim(\vec{N}) = 1$, on the other hand, this equation may be solved exactly and efficiently using any of a variety of standard numerical PDE schemes (Press et al. 1992). The Fokker–Planck approach though allows us to compare the accuracy of the methods that we will develop for the two models against the true distributions and in the results section (section ''Simulation results'') we will describe a simple illustrative example for the $\dim(\vec{N}) = 1$ case.

Forward E-step: Kalman filter. The chief advantage of the EKS algorithm is that it overcomes the computational burden of the Fokker–Planck approach, at the cost that we have to accept approximate instead of potentially exact results. As mentioned above, the basic idea is to assume that $P(\vec{N}, t)$ can be approximated by a Gaussian distribution at all times; thus we may ignore all of the details of $P(\vec{N}, t)$ and simply track the mean $\vec{\mu}_f(t) \equiv \mathrm{E}[\vec{N}(t)|O_{0,t}]$ and the covariance matrix $\sigma_f^2(t) \equiv \mathrm{Cov}[\vec{N}(t)|O_{0,t}]$ of $\vec{N}(t)$ as a function of time t. Note that $P(\vec{N}, t)$ is guaranteed to be a smooth, bounded, log-concave function of $\vec{N}(t)$ for any t whenever $P(\vec{N}, 0)$ is log-concave and the link function $f(u)$ is convex and log-concave in u (since $P(\vec{N}, t)$ may be constructed as a composition of convolutions, affine rescalings, and products with log-concave functions Paninski 2004, 2005), and therefore a Gaussian approximation will typically be justified. Figure 2 illustrates the accuracy of this approximation for the $\dim(\vec{N}) = 1$ case.

There are many different ways to construct and track this Gaussian approximation; we describe the EKS approach in depth here, and compare the advantages and limitations of several other approaches in the discussion section. The forward-step is based on the following recursion:

$$\begin{aligned} p(\vec{N}_{t+\mathrm{d}t}|O_{0,t+\mathrm{d}t}) &= \frac{p(O_{t,t+\mathrm{d}t}|\vec{N}_{t+\mathrm{d}t})}{p(O_{t,t+\mathrm{d}t})} p(\vec{N}_{t+\mathrm{d}t}|O_{0,t}) \\ &= \frac{p(O_{t,t+\mathrm{d}t}|\vec{N}_{t+\mathrm{d}t})}{p(O_{t,t+\mathrm{d}t})} \int p(\vec{N}_{t+\mathrm{d}t}, \vec{N}_t|O_{0,t})\mathrm{d}\vec{N}_t \\ &= \frac{p(O_{t,t+\mathrm{d}t}|\vec{N}_{t+\mathrm{d}t})}{p(O_{t,t+\mathrm{d}t})} \int p(\vec{N}_{t+\mathrm{d}t}|\vec{N}_t) p(\vec{N}_t|O_{0,t})\mathrm{d}\vec{N}_t. \end{aligned} \tag{15}$$

The recursive nature of the computation is clear from the fact that to compute the densities conditioned on the observations at time $t + \mathrm{d}t$ involves knowing the conditional-densities at t. In the case of Model 1, the forward-step is based on a local Taylor expansion: we first initialize the filter using Equations 13 and 14, and then we propagate $\mu_f(t)$ and $\sigma_f^2(t)$ from one time step t to the next $t + \delta t$ using the dynamics equation (this part of the forward propagation is exact in the case of Gaussian densities, due to the linearity of the dynamics (4)), next we incorporate the observed spiking data $O_{t,t+\delta t}$ approximately, via a second-order Taylor expansion of the log-likelihood, $\log p[O_{t,t+\delta t}|\vec{N}(t+\delta t)]$, about the propagated mean of $\vec{N}(t+\delta t)$. For Model 2, the main difference is that the mean of the posterior density which incorporates the observations is computed using Monte Carlo methods. A detailed derivation is presented in the Appendix.

In the case of the classical Kalman filter with continuous observations and a continuous latent process, and with Gaussian process and observational noise, the conditional densities computed at each instant are also Gaussian. Thus no approximations are made and the classical Kalman filter is optimal in a number of important senses, including the least-square sense. In the present case, however, the observational model is nonlinear, the conditional densities are not Gaussian, and we make an approximation by retaining only the mean and covariance terms.

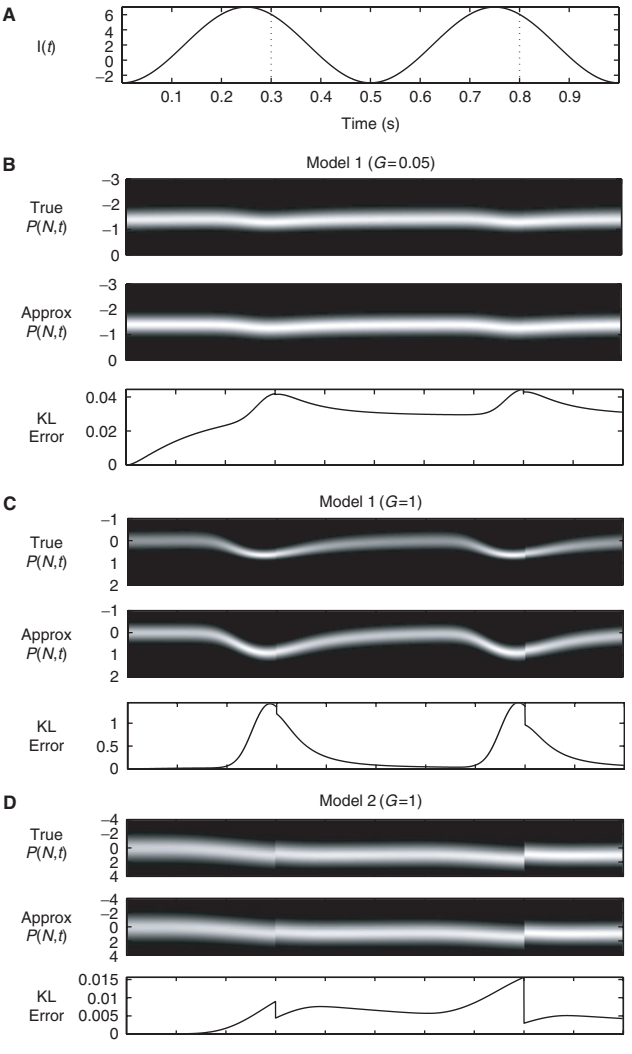

Figure 2. Comparing the Fokker–Planck approach with forward-EKS: In these simulations we numerically solve the Fokker–Planck Equations 10–12, for the case of a single neuron with a simple sinusoidal input and a one-dimensional noise-term for three different cases. The sinusoidal input in the three cases is identical and is plotted in (*A*), where the dotted vertical lines indicate the occurrence of spikes at 0.3 and 0.8 s. (B) Shows the ''true'' and the approximate probability distribution of the latent process when a small value of G is chosen ($G = 0.05$). The difference between the distributions as measured via the K–L distance is also shown. The 'true' distribution is obtained using the Fokker–Planck equations while the approximate distribution is obtained using the forward-filter derived in the text. (C) Shows similar plots for Model 1 when G is large ($G = 1$) and (D) Shows plots for Model 2 for the same large value of G chosen in (C) ($G = 1$). There are two important points to note: The first point is that the true and approximate distributions are visibly different for Model 1 in the large G case, while being similar in the small G case. The second point is that for the same large G, however, the true and approximate distributions are very similar for Model 2 as reflected by the smaller K–L distance.

For Model 1, the EKS approach reduces the complexity of the computation to $O(\mathcal{J}^3)$ per time step (we need to keep track of $O(\mathcal{J}^2)$ numbers, and updating these numbers at each time step requires $O(\mathcal{J}^3)$ time), instead of the intractable $O(S^{\mathcal{J}})$ required in the Fokker–Planck case (where S is the number of discrete points per dimension of $\vec{N}(t)$ on which we numerically solve the multivariate Fokker–Planck Equations 10–12). For Model 2, the computation also depend linearly on the sample size chosen for the Monte Carlo simulations and for a sample size of M the complexity of the computation is given by $O(M\mathcal{J}^3)$.

Forward filter, Model 1: In the case of Model 1, the convenient choice $f(\cdot) = \exp(\cdot)$ allows us to derive exact updates at spike-times, while for non-spiking updates we make a second-order approximation of the log-likelihood as considered below.

1. Dynamics update: The dynamics update step is obtained via the usual convolution representation of $\vec{N}(t)$,

$$\vec{N}(t) = \exp(-tD)\vec{N}(0) + \int_0^t \exp(-D(t-s))\vec{Z}(s)\mathrm{d}s, \tag{16}$$

giving

$$\vec{\mu}(t+\delta t)^{\dagger} = \phi\vec{\mu}_f(t), \tag{17}$$

$$\sigma^2(t+\delta t)^{\dagger} = \phi\left[\sigma_f^2(t) + \frac{\delta t}{6}\mathrm{I}\right]\phi' + \frac{\delta t}{6}\mathrm{I} + \frac{2}{3}\exp\left(\frac{-\delta tD}{2}\right)\exp\left(\frac{-\delta tD}{2}\right)\delta t, \tag{18}$$

where $\phi = \exp(-\delta tD)$ and accuracy to the second order in δt, and is obtained using Simpson's rule approximation for $\dot{\sigma}_f^2(t) = -2D\sigma_f^2(t) + \mathrm{I}$, following the discussion in (Dieci and Eirola 1994). To the first order, we have $\sigma^2(t+\delta t)^{\dagger} = \phi\sigma_f^2(t)\phi' + \delta t\mathrm{I}$.

2. Data update: For the data update step we use the fact that the product of two unnormalized Gaussian functions is also a Gaussian and that a local approximation of the matrix inverse can be obtained using

$$(A^{-1} + \epsilon B)^{-1} = A(\mathrm{I} - \epsilon BA) + o(\epsilon). \tag{19}$$

Together with a second-order expansion of the log-likelihood $\log p[O_{t,t+\delta t}|\vec{N}(t+\delta t)]$, this leads to the following updates:

$$\begin{aligned}\sigma^2(t+\delta t)^{*} &= \left[(\sigma^2(t+\delta t)^{\dagger})^{-1} + \sum_{k=1}^{K} W_k(t+\delta t)\vec{g}_k'\vec{g}_k\sigma^2(t+\delta t)^{\dagger}\right]^{-1} \\ &= \sigma^2(t+\delta t)^{\dagger}\left[\mathrm{I} - \sum_{k=1}^{K} W_k(t+\delta t)\vec{g}_k'\vec{g}_k\sigma^2(t+\delta t)^{\dagger}\right] + o(\delta t),\end{aligned} \tag{20}$$

$$\vec{\mu}(t+\delta t)^{*} = \vec{\mu}(t+\delta t)^{\dagger} - \sigma^2(t+\delta t)^{\dagger}\left[\sum_{k=1}^{K} W_k(t+\delta t)\vec{g}_k'\right] + o(\delta t), \tag{21}$$

where $W_k(t) = \exp[I_k(t) + \vec{g}_k\vec{\mu}(t)]\delta t$.

In the event that no spikes are observed from any of the cells in the time bin $[t, t+\delta t)$, we have

$$\sigma_f^2(t+\delta t) = \sigma^2(t+\delta t)^*, \tag{22}$$

$$\vec{\mu}_f(t+\delta t) = \vec{\mu}(t+\delta t)^* \tag{23}$$

If spikes are observed at time $t+\delta t$, then incorporating this into our updated state estimate, we have instead,

$$\sigma_f^2(t+\delta t) = \sigma^2(t+\delta t)^*, \tag{24}$$

$$\vec{\mu}_f(t+\delta t) = \vec{\mu}(t+\delta t)^* + \sigma^2(t+\delta t)^* G' \vec{1}_k(t), \tag{25}$$

with the $\vec{1}_k(t)$ indicating the presence or absence of a spike at time t in neuron k. As emphasized above, this update at the spike time is exact for exponential f; related (but approximate) formulas may be derived for more general link functions f.

While the forward filter does not involve any matrix inversions, we unfortunately cannot avoid a full matrix multiplication. Updates requiring a matrix inversion or full matrix multiplication each require $O(\mathcal{J}^3)$ time, which may be infeasible for very large $\mathcal{J}$.

Two speedups are readily apparent. First, it is clear that the updates between spike-times may be written as an Euler's method for an ODE solver. Thus we may use adaptive time-step methods quite fruitfully, by simply and cheaply tracking,

$$\max_{k=\{1,\ldots,K\}} W_k^*(t) = \max_k \int_u^t W_k(s)\mathrm{d}s, \tag{26}$$

where u indexes the time of the last update. On each time step, we update $\vec{W}^*(t)$, and only update $\vec{\mu}_f(t)$ and $\sigma_f^2(t)$ when $\max W_k^*(t)$ over $k = \{1, \ldots, K\}$ reaches a fixed threshold, at which point we also reset $\vec{W}^*(t)$ to zero. Our simulations show that this adaptive time-stepping can lead to a substantial speedup without significant loss of accuracy. Following the same line of reasoning, we note that not every $W_k^*(t)$ will be large. Thus, we can operate in a subspace to speed up the bottleneck computation $\sigma^2(t)\mathtt{diag}[\vec{W}^*(t)]\sigma^2(t)$. Specifically, we update $\sigma^2(t)$ only in the subspace spanned by the *significantly large* k, as measured by $W_k^*(t)$, and only set these elements of $\vec{W}^*(t)$ to zero, letting the other elements continue to integrate up to threshold.

Forward filter, Model 2: In this section we consider the forward filter for Model 2 and compute the various terms using Monte Carlo methods. Referring back to Equation 15 we have,

$$\begin{aligned} p(\vec{N}_{t+\mathrm{d}t}|O_{0,\,t+\mathrm{d}t}) &= \frac{p(O_{t,\,t+\mathrm{d}t}|\vec{N}_{t+\mathrm{d}t})}{p(O_{t,\,t+\mathrm{d}t})} \int p(\vec{N}_{t+\mathrm{d}t}|\vec{N}_t) p(\vec{N}_t|O_{0,\,t}) \vec{N}_t \\ &= \frac{p(O_{t,\,t+\mathrm{d}t}|\vec{N}_{t+\mathrm{d}t})}{p(O_{t,\,t+\mathrm{d}t})} \int p(\vec{N}_{t+\mathrm{d}t}|\vec{N}_t) \mathcal{G}_{(\vec{\mu}_f(t)\sigma_f^2(t))}(\vec{N}_t)\mathrm{d}\vec{N}_t, \end{aligned} \tag{27}$$

where we have introduced the notation $\mathcal{G}_{\vec{\mu}\sigma^2}(N)$ to denote a Gaussian distribution for the random variable N with mean $\vec{\mu}$ and covariance σ^2. The observational term is obtained from the definition of the model. For example, the probability that the k-th neuron spikes is given by,

$$p(O^k_{t,\,t+\mathrm{d}t} = 1|\vec{N}_{t+\mathrm{d}t}) = \begin{cases} \exp(V_k)\mathrm{d}t & V_k < 0 \\ (1 + V_k + V_k^2/2)\mathrm{d}t & V_k \geq 0 \end{cases} \tag{28}$$

and the probability that it does not spike is given by,

$$p(O^k_{t,\,t+\mathrm{d}t} = 0|\vec{N}_{t+\mathrm{d}t}) = \begin{cases} \exp(-\exp(V_k)\mathrm{d}t) & V_k < 0 \\ \exp(-(1 + V_k + V_k^2/2)\mathrm{d}t) & V_k \geq 0 \end{cases} \tag{29}$$

where $V_k = I_k + \vec{g}_k\vec{N}_{t+\mathrm{d}t}$ as before. The mean and covariance terms $\vec{\mu}_f(t)$ and $\sigma_f^2(t)$ are obtained recursively as described earlier. The dynamics term $p(\vec{N}_{t+\mathrm{d}t}|\vec{N}_t)$ is the same as that obtained for Model 1, since this depends only on the model for the latent process, which is similar for the two models. To compute the other terms, we resort to Monte Carlo methods with importance sampling. Many problems, including computing integrals and expectations of functions, can be formulated as Monte Carlo problems, where the true values are approximated numerically using random samples chosen from suitable distributions. For example, to compute the expectation of the function $g(\cdot)$ of a random variable N, the Monte Carlo estimate is given by

$$\mathrm{E}[g(N)] = \int g(N)p(N)\mathrm{d}N \approx \frac{1}{M}\sum_{i=1}^{i=M} g(N_i),$$

where $N_i \sim p(N)$. Thus, to compute the normalizing factor we have,

$$\begin{aligned} p(O_{t,\,t+\mathrm{d}t}) &= \int p(O_{t,\,t+\mathrm{d}t}|\vec{N}^*_{t+\mathrm{d}t})p(\vec{N}^*_{t+\mathrm{d}t})\mathrm{d}\vec{N}^*_{t+\mathrm{d}t} \\ &= \mathrm{E}_{\vec{N}^*_{t+\mathrm{d}t}}[p(O_{t,\,t+\mathrm{d}t}|\vec{N}^*_{t+\mathrm{d}t})], \end{aligned} \tag{30}$$

where $p(\vec{N}^*_{t+\mathrm{d}t})$ refers to the distribution of the latent process obtained by conditioning on the observations up to t and propagating it forward by one time-step $\mathrm{d}t$ via the dynamics equations, that is $p(\vec{N}^*_{t+\mathrm{d}t}) \sim \mathcal{G}_{(\phi\mu_f(t),\,\phi\sigma_f^2(t)\phi' + \mathcal{I}\mathrm{d}t)}(\vec{N}_{t+\mathrm{d}t})$.

We are interested in the mean and the covariance of the conditional-distribution $p(\vec{N}_{t+\mathrm{d}t}|O_{0,\,t+\mathrm{d}t})$, which can be obtained once again using Monte Carlo methods. We have,

$$\vec{\mu}_f(t + \mathrm{d}t) = \mathrm{E}_{\vec{N}_{t+\mathrm{d}t}|O_{t,\,t+\mathrm{d}t}}\left[\vec{N}_{t+\mathrm{d}t}\; p(\vec{N}_{t+\mathrm{d}t}|O_{0,\,t+\mathrm{d}t})\right]. \tag{31}$$

The covariance can be computed more quickly using,

$$\sigma_f^2(t + \mathrm{d}t) = -\left[\frac{\partial^2}{\partial\vec{N}_{t+\mathrm{d}t}}\log p(\vec{N}_{t+\mathrm{d}t}|O_{0,\,t+\mathrm{d}t})\right]^{-1}\Bigg|_{\vec{N}_{t+\mathrm{d}t}=\vec{\mu}_{f(t+\mathrm{d}t)}}. \tag{32}$$

To generate random samples we follow the 'Inverse Transform Method.' In the open interval $(0, 1)$, M equally-spaced points U_i are chosen. These points are mapped via the inverse cumulative distribution of the standard Gaussian distribution onto the real line to obtain our Gaussian random sample $(Z_i = \Phi(U_i)^{-1})$. These random samples from the standard Gaussian distribution are generated once and for all at the beginning of the simulation and are used in the Monte Carlo methods via the transformation $X_i = \sqrt{C}Z_i + \mu$ when samples from a Gaussian distribution with a mean μ and covariance C are required.

The computational cost of the Monte Carlo method for the forward filter scales linearly with the sample size. Our simulations indicate that, in the parameter range of interest, a sample size of the order of 10^2 is necessary to obtain good estimates, resulting in this approach taking two orders of magnitude more time than the forward-filter for Model 1. Also we note that the M-step requires the computation of certain expectations, as can be seen from Equation 9. For Model 2, these expectations are also computed using Monte Carlo methods, and as this is a part of the maximization step, these integral need to be computed several times during each iteration of the EM algorithm.

Backward E-step: Kalman smoother. The backward-sweep of the E-step for hidden Markov processes uses the output of the forward-step, $p(\vec{N}(t)|O_{0,t})$, in order to compute the fully-conditioned distributions, $p(\vec{N}(t)|O_{0,T})$, and $p(\vec{N}(t), \vec{N}(t+\mathrm{d}t)|O_{0,T})$. The algorithm for the backward filter can be derived using the Bayesian approach. It is initialized by noting that the forward-step gives the fully-conditioned distribution at time T and by considering,

$$\begin{aligned} p(\vec{N}_{T-\mathrm{d}t}|O_{0,T}) &= \frac{1}{p(O_{0,T})} p(\vec{N}_{T-\mathrm{d}t}, O_{0,T}) \\ &= \frac{1}{p(O_{0,T})} \int p(\vec{N}_{T-\mathrm{d}t}, \vec{N}_T, O_{0,T}) \mathrm{d}\vec{N}_T \\ &= \frac{1}{p(O_{0,T})} p(\vec{N}_{T-\mathrm{d}t}|O_{0,T-\mathrm{d}t}) \int p(O_T|\vec{N}_T) p(\vec{N}_T|\vec{N}_{T-\mathrm{d}t}) \mathrm{d}\vec{N}_T. \end{aligned}$$

The backward filter smoothes the conditioned distribution by incorporating the dynamics of the hidden process backwards in time, and as such is independent of the observational model. As our observational model is Gaussian, this computation is standard (Mendel 1995; de Jong and MacKinnon 1988; Smith and Brown 2003). Denoting the fully conditioned statistics by $\vec{\mu}_s(t) \equiv \mathrm{E}[\vec{N}(t)|O_{0,T}]$, $\sigma_s^2(t) \equiv \mathrm{E}[\vec{N}(t)\vec{N}(t)'|O_{0,T}]$ we have

$$\vec{\mu}_s(t) = \vec{\mu}_f(t) + A(t)(\vec{\mu}_s(t+\mathrm{d}t) - \phi\vec{\mu}_f(t)), \tag{33}$$

$$\sigma_s^2(t) = \sigma_f^2(t) + A^2(t)(\sigma_s^2(t+\delta t) - (\phi\sigma_f^2(t)\phi' + \mathrm{I}\delta t)), \tag{34}$$

and

$$\mathrm{E}[\vec{N}(t)\vec{N}'(t+\delta t)|O_{0,T}] = A(t)\sigma_s^2(t+\delta t) + \vec{\mu}_s(t)\vec{\mu}_s'(t+\delta t), \tag{35}$$

where $A(t) = \sigma_f^2(t)\phi'[\phi\sigma_f^2(t)\phi' + \delta t\mathrm{I}]^{-1}$. We may use the usual mean-variance decomposition to obtain

$$\mathrm{E}[\vec{N}(t)\vec{N}'(t)\big|O_{0,T}] = \vec{\mu}_s(t)\vec{\mu}_s'(t) + \sigma_s^2(t). \tag{36}$$

We use the notation $\vec{\mu}_s(t)$ and $\sigma_s^2(t)$ here to emphasize that the means and covariances computed via the full forward-backward method are typically smoother functions of time than the forward means $\vec{\mu}_f(t)$ and covariances $\sigma_f^2(t)$ (Figure 3).

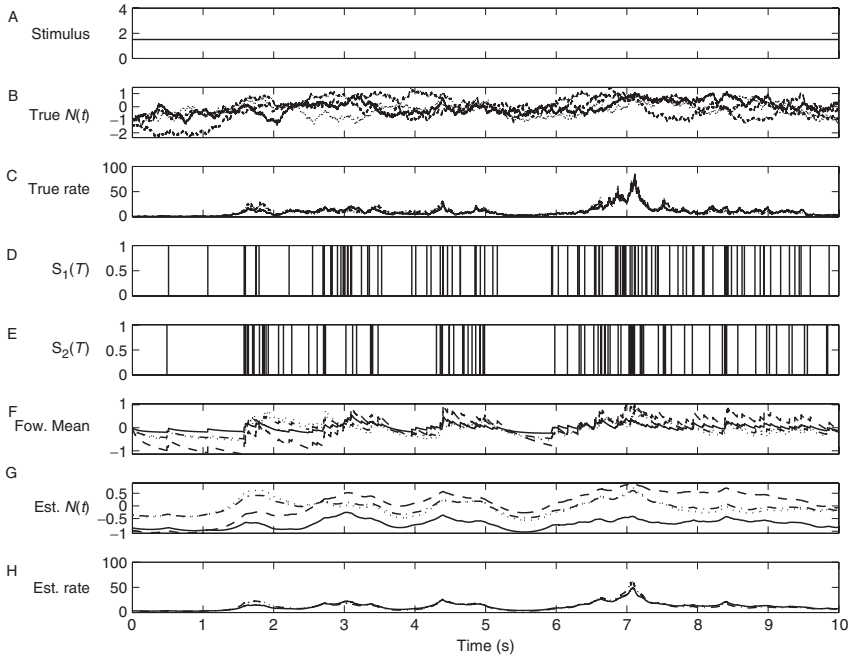

Figure 3. Example of forward–backward Kalman smoother: In this simulation the firing-rates of two cells was modulated by: A constant input, the neuron's post-spike waveform, interneuronal cross-coupling terms, and the latent OU processes as given by Equation 42. (A) Shows the constant d.c. input. (B) Shows the true values for the four OU processes used to generate data on a single trial. (C) Shows the true intensity function for both the cells. Cell one is indicated by solid line and cell two by dotted line. (D) and (E) Shows the spike data generated in a single trial from the the two cells. (F) Shows the conditional mean of $\vec{N}(t)$ computed via the forward filter given the true values of the parameters, the stimulus, and the spike-data from $[0, t)$. Note the upward jumps in mean corresponding to spike-times followed by continuous decay. Conditional means for both cells jump simultaneously, though with amplitude depending on which neuron was observed to spike, as can be seen from Equation 25. The backward filter (G) estimates $\vec{N}(t)$ by conditioning on the entire data sample. Note that the mean obtained from the full spike train is continuous, unlike the forward mean. (H) shows the estimated firing rate of the two cells obtained from the smoothed estimate of $\vec{N}(t)$ along with the known stimulus by using the moment-generating function for the Gaussian distribution (40). The values of the various parameters used in this simulation are noted in the accompanying text.

The Maximization step

In the M-step we need to maximize the expected value of the complete data log-likelihood given in Equation 9 with respect to θ to obtain the parameter estimates for the next iteration of the EM algorithm. From Equation 9 we see that the M-step for the dynamics-parameter D decouples from that of the intensity-parameters $\{G, \{\vec{r}_k\}, \{h_{kj}\}\}$. In this section we consider the implementation of the M-step for the two models to determine these parameters.

Dynamics-parameter. From Equation 9 we see that this part of the M-step involves finding $\mathrm{E}[\log p(\vec{N}_{0,T}|\theta)|O_{0,T}, \theta_{i-1}]$. Since the latent process for the Models 1 and 2 is identical, this part of the M-step is the same for the two models. To determine this term we need to compute the following expectations,

$$
\begin{aligned}
&\mathrm{E}\Big[\log[p(\vec{N}_{0,T}|\theta)\big|O_{0,T}, \theta_{i-1}]\Big] \\
&\quad = \mathrm{E}\Big[\log[p(\vec{N}(0)|\theta)] + \sum_{t=\delta t}^{t=T} \log[p(\vec{N}(t)|\vec{N}(t-\delta t), \theta)]\Big| O_{0,T}, \theta_{i-1}\Big] \\
&\quad = \mathrm{E}[-\frac{1}{2}\vec{N}(0)'\Big(\frac{1}{2}D^{-1}\Big)^{-1}\vec{N}(0)] \\
&\qquad + \mathrm{E}\Big[\sum_{t=\delta t}^{t=T} -\frac{1}{2\delta t}\Big[(\vec{N}(t) - \phi\vec{N}(t-\delta t))'(\vec{N}(t) - \phi\vec{N}(t-\delta t))\Big]\Big| O_{0,T}, \theta_{i-1}\Big] \\
&\quad = \Big[1 + O\Big(\frac{1}{T}\Big)\Big]\mathrm{E}\Big[\sum_{t=\delta t}^{t=T} -\frac{1}{2\delta t}\big[(\vec{N}(t) - \phi\vec{N}(t-\delta t))'(\vec{N}(t) - \phi\vec{N}(t-\delta t))\big]\Big| O_{0,T}, \theta_{i-1}\Big],
\end{aligned}
\tag{37}
$$

where again $\phi = \exp(-\delta t D)$, and we have neglected, in the final step, the contribution to the expectation by the initial condition $P(\vec{N}, 0)$. The expectation has taken over the conditional measure of $\vec{N}(t)$ given the current parameter settings and the observed spike data on $[0, T]$. The output of the forward-backward algorithm outlined in the earlier sections allow us to compute the various terms in this expectation. Differentiating Equation 25 with respect to ϕ we obtain ϕ^{new} for the next iteration,

$$
\phi^{\mathrm{new}} = \Big(\sum_{t=\mathrm{d}t}^{T} \mathrm{E}[N(t)N(t-\mathrm{d}t)'|O_{0,T}, \theta_{i-1}]\Big)\Big(\sum_{t=\mathrm{d}t}^{T} \mathrm{E}[N(t)|O_{0,T}, \theta_{i-1}]\Big)^{-1}. \tag{38}
$$

We obtain the elements of D from $D = -(1/\delta t)\log(\phi)$.[1]

[1]In our experience, the resulting estimate of D always corresponds to a stable dynamical system (i.e., the eigenvalues of $D+D'$ are negative), and therefore our constraint on D has been satisfied automatically. If this is not the case, a straightforward expansion of ϕ in δt reduces Equation 37 to a quadratic function of D, up to $o(\delta t)$ terms; maximizing this quadratic function under the stability constraints is now a semi-definite program (Boyd and Vandenberghe 2004), to which standard convex optimization procedures may be applied.

M-step: intensity-parameters. To obtain the values for the intensity-parameters $\{G, \{\vec{r}_k\}, \{h_{kj}\}\}$ we need to maximize the other term from Equation 9 with respect to the parameters, that is maximize $\mathrm{E}[\log p(O_{0,T}|\vec{N}_{0,T}, \theta)|O_{0,T}, \theta_{i-1}]$ with respect to θ. This term will be different for the two models.

Model 1: We have,

$$\begin{aligned}
&\mathrm{E}\left[\log[p(O_{0,T}|\vec{N}_{0,T}, \theta)\middle| O_{0,T}, \theta_{i-1}\right] \\
&= \sum_{k=1}^{K} \mathrm{E}\left[\sum_j (I_k(t_{k,j}) + \vec{g}_k\vec{N}(t_{k,j})) - \int_0^T \exp(I_k(s) + \vec{g}_k\vec{N}(s))\mathrm{d}s\middle| O_{0,T}, \theta_{i-1}\right] \\
&= \sum_{k=1}^{K}\left[\sum_j (I_k(t_{k,j}) + \vec{g}_k\vec{\mu}_s(t_{k,j})) - \int_0^T \exp(I_k(s))\mathrm{E}[\exp(\vec{g}_k\vec{N}(s))\mathrm{d}s|O_{0,T}, \theta_{i-1}]\right] \\
&= \sum_{k=1}^{K}\left[\sum_j (I_k(t_{k,j}) + \vec{g}_k\vec{\mu}_s(t_{k,j})) - \int_0^T \exp(I_k(s))\exp[\vec{g}_k\vec{\mu}_s(s) + \frac{1}{2}\vec{g}_k\sigma_s^2(s)\vec{g}_k']\mathrm{d}s\right],
\end{aligned} \tag{39}$$

where we have used the fact that neurons in the ensemble are conditionally independent given the latent process, along with the moment-generating function for the multivariate normal distribution,

$$\mathrm{E}_{\vec{z}\sim\mathcal{N}(\vec{\mu},\sigma^2)}[\exp(\vec{y}'\vec{z})] = \exp\left(\vec{y}'\vec{\mu} + \frac{1}{2}\vec{y}'\sigma^2\vec{y}\right). \tag{40}$$

In Equation 39 we have a positively-weighted generalization of the usual point-process likelihood (Smith and Brown 2003; Paninski 2004); the key fact is that this function is jointly concave as a function of the parameters $(\vec{g}_k, \vec{r}_k, \{h_{kj}\})$; thus we may compute gradients with respect to the parameters and carry out the maximization via straightforward conjugate gradient ascent. Note, therefore, that this M-step reduces to K smaller independent optimization problems, all of which have unique global optimizers which can easily be computed in parallel by standard conjugate gradient ascent algorithms.

Model 2: We have,

$$\begin{aligned}
&\mathrm{E}\left[\log[p(O_{0,T}|\vec{N}_{0,T}, \theta)\middle| O_{0,T}, \theta_{i-1}\right] \\
&= \sum_{k=1}^{K} \mathrm{E}\Bigg[\sum_{j:V_k(t_{k,j})<0} (I_k(t_{k,j}) + \vec{g}_k\vec{N}(t_{k,j})) + \sum_{j:V_k(t_{k,j})\geq 0} (\log(1 + V_k(t_{k,j}) + V_k(t_{k,j})^2/2) \\
&\quad - \int_{0:V_k(s)<0}^{T} \exp(I_k(s) + \vec{g}_k\vec{N}(s))\mathrm{d}s - \int_{0:V_k(s)\geq 0}^{T} (1 + V_k(s) + V_k(s)^2/2)\mathrm{d}s\middle| O_{0,T}, \theta_{i-1}\Bigg].
\end{aligned} \tag{41}$$

The expectations required of the M-step of Model 2, as given by Equation 41, are computed using Monte Carlo methods. We can also evaluate the gradients, of this expectation, with respect to the parameters using Monte Carlo methods; allowing us to ascend this function using gradient-ascent methods.

We have found that a good initialization of the parameters $\{r_k, h_{jk}(.)\}$ (for the first M-step) is provided by the the optimizer of the classical loglikelihood, that is, with G set to zero. For all subsequent M-steps, the optimizer $\hat{\theta}_{i-1}$ from the last iteration is used as the initialization to the ascent algorithm. Finally, as usual, prior information in the form of a log-concave prior distribution on the parameters θ, $p(\theta) = \exp(-Q(\theta))$, for some convex "penalty" function $Q(\theta)$, may be easily incorporated: to compute the maximum a posterior (MAP) estimate for θ instead of the maximum likelihood estimator, we simply ascend the functions (37) and (39 and 41) added to the log-prior term $- Q(\theta)$; the usual proof that EM ascends the log-likelihood surface applies unchanged to establish that this penalized EM ascends the log-posterior surface, i.e., that penalized EM returns the MAP estimate for θ.

Simulation results

In this section we explore the system identification tools developed above by considering several modeling scenarios.

Fokker–Planck simulations

In our first set of simulations we consider the error, introduced by the Gaussian approximation, in our inference of the forward conditioned-distribution of the latent process for the two models. That is, we compare the solution of the forward-EKS step with the numerical solution obtained from solving the Fokker–Planck equation. To simulate the Fokker–Planck equation we consider the simplest possible model of a single neuron, with a scalar $N(t)$, no spike history terms and instantaneous stimulus filtering. As $N(t)$ is one-dimensional, we can solve the Fokker–Planck Equations 10–12 numerically in order to compare the *exact* solution to the approximate forward-EKS solution. The results of this comparison are shown in Figure 2. We compare a small-noise and a large-noise case, with G set to 0.05 and 1.0, respectively for Model 1. We also compute the true and approximate distributions for Model 2 for the large-noise case with $G = 1$ as before. The dynamics term $D = 2$ and the stimulus filter $\vec{r}_k = 1$ in each case. $x(t)$ is chosen to be a simple sinusoid. We compare $P(N, t)$ given this input and two spikes at times $t = 0.3$ and 0.8.

We immediately see that for Model 1, the EKS solution is rigorously valid only in the small G limit. This is seen from considering the derivation of the forward-EKS (presented in the Appendix) for the non-spike updates, where a local Taylor expansion is used to approximate the likelihood. This expansion is reasonably accurate only when the $\tilde{N}$ term is small. Figure 4 makes a similar point via simulations. We see that the forward-EKS approximation is quite good in the small-noise case but only qualitatively accurate in the case of larger noise. Though $P(N, t)$ is well approximated by a Gaussian for all times t, the mean and variance of the forward-EKS approximation are significantly off, as can be quantified by computing the Kullback–Leibler divergence between the

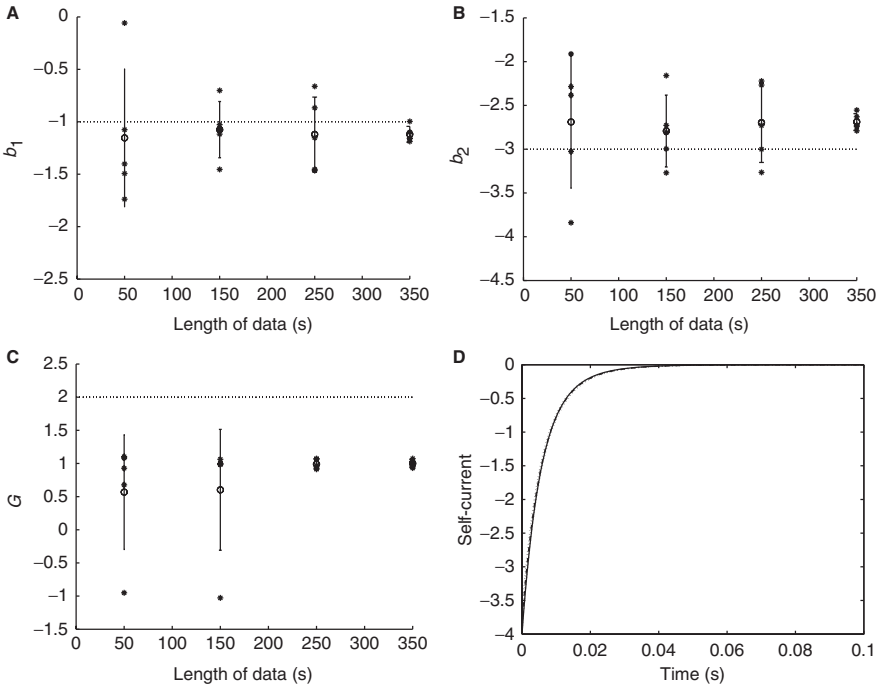

Figure 4. System Identification for Single Neuron (Model 1): The true parameter values for b_1, b_2 and G (dotted line) and the estimates obtained from data of different durations $T = \{50, 150, 250, 350\}$ seconds are plotted in (A), (B), and (C). For each duration T the spike-data is generated from the model given by Equation 44, and the parameters are identified. These estimates are indicated by asterisks and the mean of the estimate by circles. The variance of the estimates decreases as the length of the data increases. G is consistently underestimated here. (D) Shows that the true post-spike inhibitory self-current (solid line) is within the 95% confidence limits (dashed lines) about the estimated post-spike current (dash-dot line) obtained using the mean and the variance of the estimates for b_1 and b_2 from data-sample of duration 350 s. Note that the self-inhibitory currents are estimated accurately despite the bias in estimate of G.

true and approximate solutions for $P(N, t)$. Since the EM algorithm is based on the calculation of $P(N, t)$ we can expect that the EM algorithm will lead to more accurate estimates in the presence of small G terms. We will examine this phenomenon in further detail subsequently.

However, in the case of Model 2, where we have truncated the exponential term of the link function, we see that the true and approximate distributions are very similar even for the value of $G = 1$ where Model 1 showed a larger error. The fact that the true distribution is better inferred by Model 2 implies that the EM algorithm will be better able to estimate the parameters of this model, a point that we will return to in further simulations subsequently.

Extended Kalman smoother simulation

The forward-EKS step computes the first two moments of the latent process at an instant in time, conditioned on the observations made up to that time,

while the forward–backward EKS computes the moments conditioned on the entire data.

In Figure 3 we demonstrate the forward and forward–backward steps of the EKS implemented for Model 1. The two cells in this simulation receive a constant stimulus input of $\vec{r}_k \cdot \vec{x}(t) = 1.5$ and common inputs from four OU processes with time-constants 0.5 s, 1 s, 1 s, and 2 s. The post-spike waveform reflecting the refractory nature of a spiking neuron is taken to be the weighted sum of two exponentials with time-constants $\tau_1 = 2\,\text{ms}$ and $\tau_2 = 5\,\text{ms}$, while the interneuronal cross-coupling term is taken to be the weighted sum of two exponentials with time-constants $\tau_3 = 10\,\text{ms}$ and $\tau_4 = 50\,\text{ms}$. The intensity function of neuron k is therefore given by

$$\lambda_k(t) = \exp\left[1.5 + \sum_{\{j:t_{j,k}<t\}} B\begin{bmatrix}\exp\left(-\frac{t_{j,k}}{\tau_1}\right)\\ \exp\left(-\frac{t_{j,k}}{\tau_2}\right)\end{bmatrix} + \sum_{i=1,\,i\neq k}^{K}\sum_{\{j:t_{j,i}<t\}} C\begin{bmatrix}\exp\left(-\frac{t_{j,i}}{\tau_3}\right)\\ \exp\left(-\frac{t_{j,i}}{\tau_4}\right)\end{bmatrix} + \vec{g}_k\vec{N}(t)\right], \tag{42}$$

where the matrices B, C, and G used in the simulation are given by

$$B = 10^{-2}\begin{bmatrix}-2 & -2\\ -3 & -5\end{bmatrix}, \quad C = 10^{-2}\begin{bmatrix}2 & 3\\ 1 & -2\end{bmatrix} \quad G = 10^{-1}\begin{bmatrix}8 & 5 & 7 & 5\\ 5 & 10 & 6 & 8\end{bmatrix}. \tag{43}$$

The forward and backward steps of the EKS are useful in solving several inference tasks. For example, given the stimulus and the model we can compute various quantities of interest, such as the mean, variance, and auto- and cross-correlations of the firing rate, by simply running Equation 4 forward and using Equation 1 to recursively generate spikes. The forward filter also allows us to compute the likelihood of a given observed spike-train under the assumed model.

The EKS also enables us to infer the activity of a group of cells in a network given the activity of another group of disjoint cells. Again considering Model 1, an example of this application is shown in Figure 5, where the activity of ten cells is modulated by a four-dimensional OU process. Only nine of these cells are observed, and the spiking data obtained from these nine cells is used to predict the spike rate of the tenth unobserved cell.

EM simulations

In this subsection we analyze the performance of the EM algorithm, applied to several models of increasing complexity.

Single neuron with refractoriness: Model 1 and Model 2. In these set of simulation we compare the accuracy of the estimates obtained using the EM algorithm for Models 1 and 2. We consider a spike train obtained from a single neuron whose

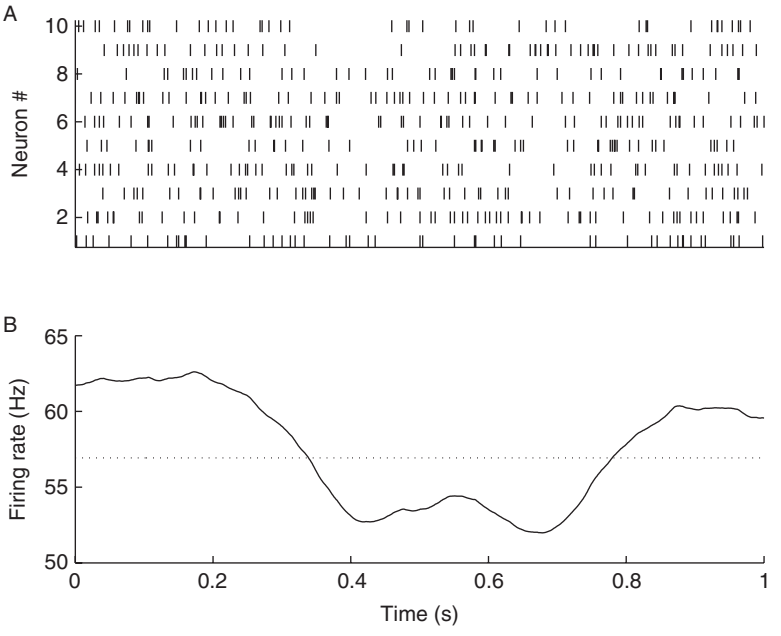

Figure 5. Example of inference using the EKS: Here we simulated the activity of ten cells with a constant input stimulus of unit amplitude, and correlated noisy-input arising from the activity of a four-dimensional OU process with time constants of 0.5, 1, 2, and 4 s. For simplicity, in this simulation, we assume that there are no cross-coupling terms. The true parameters are assumed to be known. (A) Shows the true spiking activity of the ten neurons. The spike-trains of neurons 1–9 are used in the EKS to estimate the latent process, which in turn is used to obtain the estimated spike-rate of the unobserved neuron #10. (B) Shows the estimated spike rate of neuron #10 using our knowledge of the activity of the other cells in the network and the stimulus (solid line) and using only the stimulus (dotted line).

spike-rate is modulated by a known constant stimulus, inhibitory currents which peak every time the neuron spikes (capturing the refractoriness of the neuron), as well as a one-dimensional noise-driven input. We have

$$V(t) = \left[2 + \sum_{i=1}^{2} \sum_{\{j:t_j<t\}} b_i \exp\left(-\frac{t - t_j}{\tau_i} \right) + GN(t) \right], \tag{44}$$

where the firing-rate for Model 1 is given by Equation 5 and for Model 2 is given by Equation 6, and where t_j refers to the time of the j-th spike of the neuron. Though we consider the effect of the refractory terms to be the sum of exponentials with different decay rates, the formulation is general and allows for other kernels. The time-constant τ of the OU process was taken to be 2 s (modeling slow fluctuations in the underlying mean firing rate), while the time-constants for the exponentials were taken to be 10 ms and 5 ms. We generate the spiking data by recursively sampling from Equation 44, and we then subsequently fit the model to the data using the EM algorithm. For this as well as all the subsequent simulations we fix $\delta t = 5 \times 10^{-4}$. A large value of G

($G = 2$) was used in these set of simulations. System identification was carried out multiple times over independent, identically distributed data samples of different lengths T as shown in Figure 4.

We can see from Figure 4 that for Model 1, G was significantly underestimated. In the next set of simulations (Figure 6), for the same single neuron (Model 1), we explore the effect of the size of G in estimating the parameters accurately. As the Fokker–Planck simulations (Figure 3) indicate, the accuracy of the estimates decreases as G increases. This bias in our estimation of G is not seen for Model 2 as can be seen from the Figure 7. Note however that for both the models the bias in the estimates of b_1 and b_2 was substantially less, and the self-current $\sum_i b_i \exp(-t/\tau_i)$ was estimated accurately as well.

The computational effort required for estimating the parameters in this simple case for Model 2 was orders of magnitude greater than for Model 1. Thus, though the parameters are estimated accurately, the computational burden is greater than for Model 1, and in our subsequent simulations, which are more detailed than the present case, we present results only for Model 1.

Single neuron with realistic stimulus filter. In the next set of simulations we consider the problem of identifying a more complicated stimulus filter. Spiking data is generated for a single neuron with refractoriness and with a receptive field which is similar to the classical spike-triggered receptive field shape given in Figure 2.14 of (Dayan and Abbott 2001). The firing rate of the neuron is given by

$$\lambda(t) = \exp\left[a_1 + \sum_{i=2}^{i=6} a_i \cos(2\pi i t/T_0) - \sum_{i=7}^{i=11} a_i \sin(2\pi i t/T_0) + \sum_{i=1}^{2} \sum_{\{j:t_j<t\}} b_i \exp\left(-\frac{t-t_j}{\tau_i}\right) + GN(t)\right], \tag{45}$$

where $T_0 = 0.3\,\mathrm{s}$ and a_i are the coefficients of the receptive field in the Fourier domain. Figure 8 shows a plot of the true vs. estimated receptive field, along with a plot for the post-spike inhibitory currents. The values of the true and estimated parameters, along with the standard deviations of the estimates, are presented in Table I; the estimates are seen to be fairly accurate here.

Several interconnected neurons with common input. In this final set of simulations we consider a richer model with four neurons where the firing rate of neuron k at the time t is given by,

$$\lambda_k(t) = \exp\left[4 + \sum_{i=1}^{2} \sum_{\{j:t_{j,k}<t\}} b_{(k,i)} \exp\left(-\frac{t-t_{j,k}}{\tau_i^{\mathrm{self}}}\right) + \sum_{i=1}^{2} \sum_{l=1,l\neq k}^{K} \sum_{\{j:t_{j,l}<t\}} c_{(k,i)} \exp\left(-\frac{t-t_{j,l}}{\tau_i^{\mathrm{cross}}}\right) + g_k \vec{N}(t)\right], \tag{46}$$

where $\vec{N}(t)$ is a two-dimensional OU process. This formulation captures the effect on the firing rate of the stimulus through the constant term, the refractory terms are

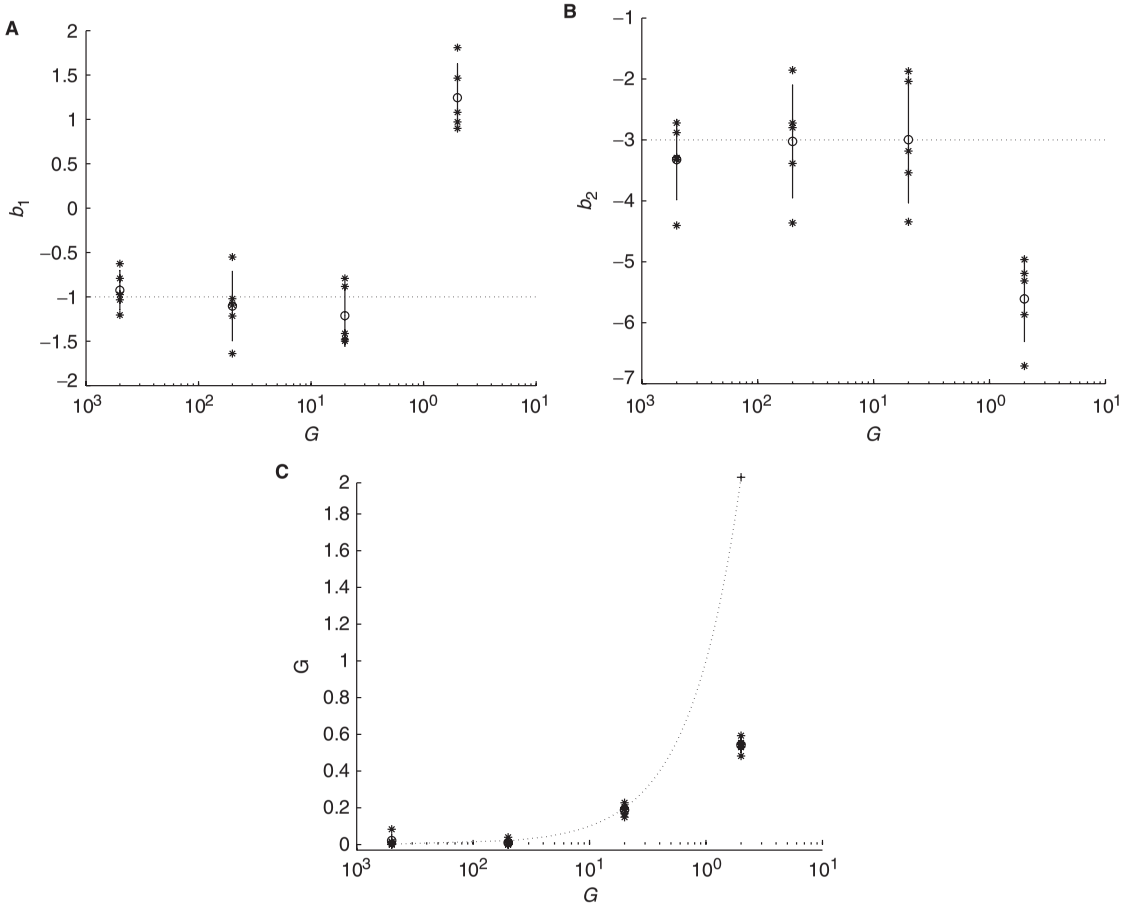

Figure 6. Effect of G on the accuracy of the estimates (Model 1): In this set of simulations we explore the effect of G on the accuracy of the estimates of the parameters for Model 1, for the simple case of a single neuron whose firing rate is given by Equation 44. (A), (B), and (C) These estimates are indicated by asterisks and the means by circles of b_1, b_2, and G; for true values of $G = \{0.002, 0.02, 0.2, 2\}$ denoted by '+'. As the Fokker–Planck simulations (Figure 3) indicate, the accuracy of the estimates decreases for larger G.

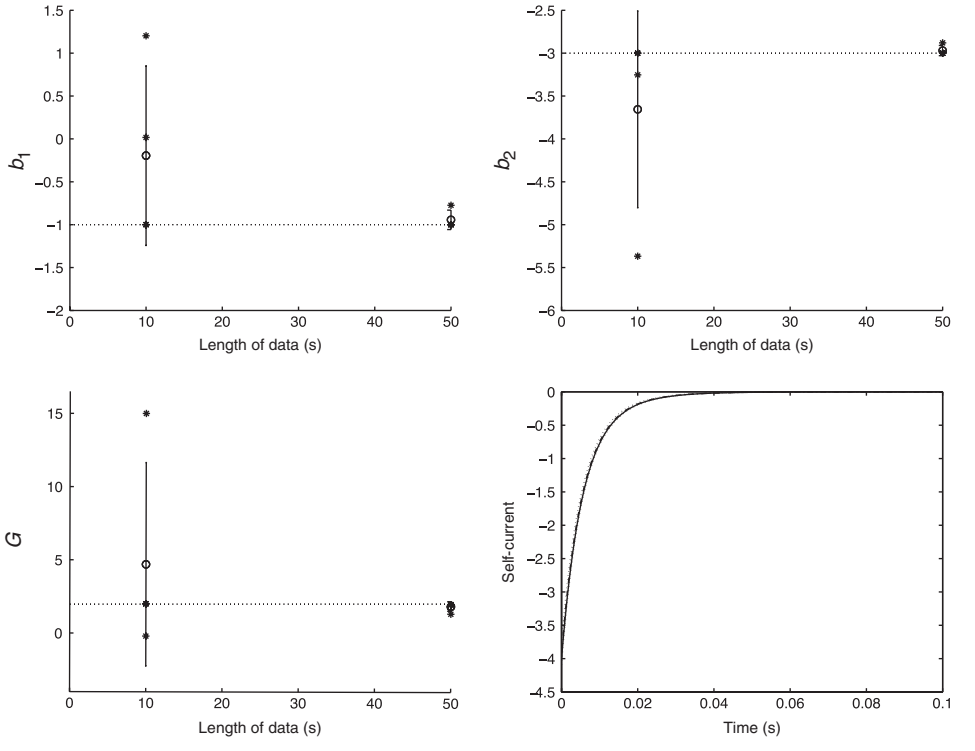

Figure 7. System identification for single neuron (Model 2): The true parameter values for b_1, b_2 and G (dotted line) and the estimates obtained for Model 2 from data of different durations $T = \{10, 50\}$ seconds are plotted in (A), (B), and (C). As in Figure 4, for each duration T the spike-data is generated from the model given by Equation 44, and the parameters are identified. These estimates are indicated by asterisks and the means by circles. The variance of the estimates decreases as the length of the data increases. (D) Shows that the true post-spike inhibitory self-current (solid line) is within the 95% confidence limits (dashed lines) about the estimated post-spike current (dash-dot line) obtained using the mean and the variance of the estimates for b_1 and b_2 from data-sample of duration 50 s. The point to be noted here is that the estimated G is closer to the true value than the estimate for Model 1 (Figure 6). Also note that Model 2 requires less data to estimate the parameters of the model accurately.

captured through b and the inputs due to the firing of other neurons is captured through c. The estimated and true parameter values, along with the standard deviations of the estimates, are tabulated in Table II (the duration T was taken to be 450 s here); again, we see that the parameters are accurately recovered.

Discussion

In this article we have considered the problem of estimating the connectivity of neurons in a network using observed spike-train data and the stimulus presented to the network, while accounting for the effect of unobserved neurons on the firing of the observed neurons. The effect of these unobserved neurons is modeled as a latent process modulating the firing activity of the observed neurons. Our model

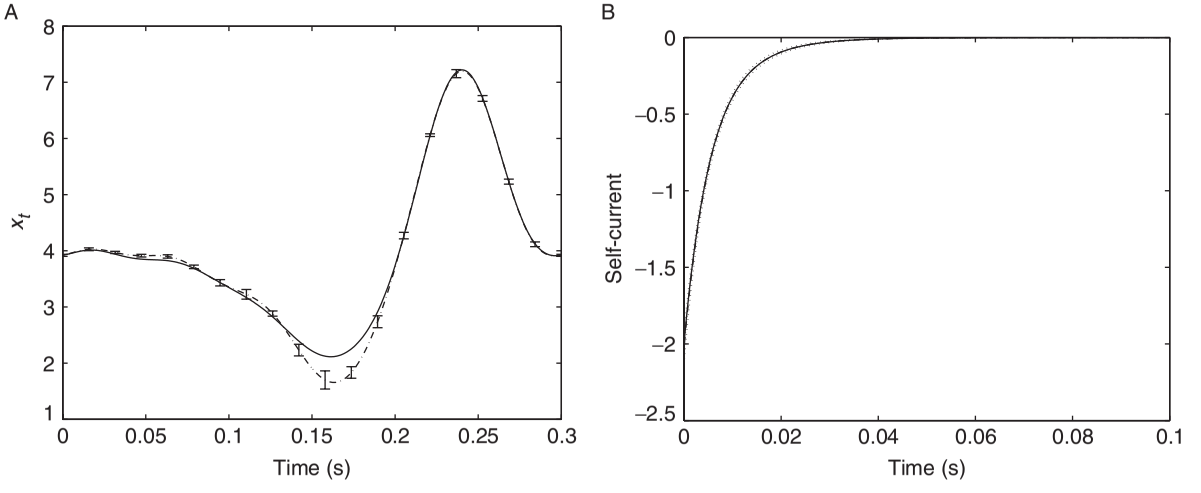

Figure 8. Estimating the receptive field: This simulation identifies a neuron with a receptive field similar to the classical spike-triggered receptive field (45). (A) shows the shape of the true receptive field (solid line) and the estimate (dash-dot line). (B) shows the true (solid line) and the estimated (dash-dot line) post-spike inhibitory current along with the 95% confidence interval (dotted lines).

Table I. Estimated and true parameter values for neuron with realistic stimulus filter.

Parameter	True	Estimate	Parameter	True	Estimate
G	0.20	0.18 ± 0.02	a_5	0.08	0.06 ± 0.03
b_1	-0.50	-0.49 ± 0.11	a_6	0.08	0.09 ± 0.04
b_2	-1.50	-1.54 ± 0.17	a_7	0.96	0.90 ± 0.011
a_1	4.00	3.95 ± 0.017	a_8	0.56	0.62 ± 0.018
a_2	1.09	1.20 ± 0.04	a_9	-0.43	-0.51 ± 0.03
a_3	-1.01	-1.10 ± 0.04	a_{10}	-0.20	-0.14 ± 0.018
a_4	-0.32	-0.29 ± 0.04	a_{11}	-0.04	-0.07 ± 0.013

Table II. Estimated and true parameter values for several interconnected neurons.

Parameter	True	Estimate	Parameter	True	Estimate
$b_{1,1}$	-1.00	-1.33 ± 0.08	$b_{1,2}$	-2.00	-1.69 ± 0.05
$b_{2,1}$	-0.50	-0.69 ± 0.10	$b_{2,2}$	-2.50	-2.32 ± 0.08
$b_{3,1}$	-3.00	-3.20 ± 0.06	$b_{3,2}$	-1.00	-0.82 ± 0.04
$b_{4,1}$	-2.00	-2.23 ± 0.06	$b_{4,2}$	-1.00	-0.76 ± 0.04
$c_{1,1}$	0.20	0.17 ± 0.02	$c_{1,2}$	0.10	0.12 ± 0.01
$c_{2,1}$	0.30	0.29 ± 0.02	$c_{2,2}$	0.15	0.16 ± 0.03
$c_{3,1}$	0.20	0.20 ± 0.03	$c_{3,2}$	0.30	0.29 ± 0.03
$c_{4,1}$	0.18	0.11 ± 0.002	$c_{4,2}$	0.20	0.26 ± 0.006

formulation is sufficiently general and allows us to consider several other important modeling scenarios. For example, to consider the important situation where the unobserved neurons are influenced by the stimulus, we can introduce a stimulus dependent term in the dynamics equation for the latent process, and estimate it using the tools developed in this article. We can also consider the effect of adaptation, which is observed typically in the response of neural circuits to repeated stimulation, by incorporating a term similar to the self-inhibitory that we considered but which acts on longer time-scales.

Here, we have considered two models and developed methods to estimate the parameters of the network by following the state-space approach to the modeling of point-process observations (Smith and Brown 2003). We have used a modified computationally tractable EM algorithm, where the E-step is carried out by using forward and backward sweeps of the EKS, and the M-step involves the maximization of concave functions. The state-space approach to modeling point-processes has received increasing attention recently. The other work (aside from that of Smith and Brown (2003) that most closely corresponds, in methodology, to what we have presented in this article is that of Yu et al. (2005), who modeled the hidden dynamics of a behavioral motor task using a nonlinear extended Kalman framework. As in our work, the key idea in Yu et al. (2005) is that the activity of the observed population of neurons is modulated by some common unobserved dynamical process. The main difference is in the interpretation of this underlying dynamical process: in Yu et al. (2005), this process corresponds to the evolution of a decision to move in a delayed-reach task, while in our case the process corresponds to a noisy lumped sum of activity from a population of presynaptic neurons. Another

(less important) difference is that the underlying dynamics in our model were chosen to be linear, implying that many of the steps in the EKS algorithm could be implemented exactly (because linear dynamics preserve Gaussianity), and allowing us to make use of straightforward adaptive time-stepping ideas to improve the numerical efficiency of the algorithm (recall section ''Forward E-step: Kalman filter''), whereas the nonlinear hidden dynamics used in Yu et al. (2005) require further approximations to apply the EKS algorithm (Wan and Merwe 2004). In addition, we have emphasized the incorporation of the coupling terms $\{h_{kj}(.)\}$, which allow us to capture refractoriness and burstiness effects as well as direct network coupling effects (these terms could be easily incorporated in the framework of Yu et al. (2005) as well).

While Yu et al. (2005) consider a different modeling problem using similar tools, Nykamp (2005, 2007) considers a similar problem (inferring unobserved common input terms from the observed activity of a population of possibly coupled neurons) using methods which are significantly different from ours. The approach adopted in Nykamp (2005, 2007) is to use the joint stimulus-spike statistics to determine the neuronal connectivity via a version of the method of moments, or alternatively via an expansion in the coupling terms of the point-process likelihood of the fully-observed neural population. This expansion analysis is carried out under the assumption that the inter-neuronal coupling is weak, similar in some respects to our assumption that G is small in the case of Model 1 (which ensures that our Gaussian approximation is reasonable), although in our case we do not need to assume that the coupling terms $\{\vec{r}_k\}$ or $\{h_{kj}(.)\}$ are small. An important direction for future research is to further compare the strengths and limitations of these two approaches.

The key limitation of our approach, for Model 1, is the bias in the estimates of the parameters when the modulation of the firing rate by the $\vec{N}(t)$ terms is high. This bias arises, because while incorporating the observations in our estimate of the state in the derivation of the forward-EKS step (Appendix), we approximate the posterior distribution of the state by considering a second-order Taylor series approximation about the mean of the prior distribution of $\vec{N}(t)$. We decided to pursue the EKS approach here primarily due to its computational efficiency. On the other hand, we considered Monte Carlo methods for Model 2, where we have accurate parameter estimation but we have to contend with the excessive computational burden of this approach.

Different strategies can be adopted to overcome both these limitations. For one, the algorithm can be speeded up significantly by resorting to *distributed computing*, where the problem is broken into smaller independent problems which can be farmed out to a bank of processors. For example for Model 1, the M-step as mentioned in the text, can be split into a number of smaller independent optimization problems. In the case of the Monte Carlo approach for Model 2, each of the processors can consider a fraction of the random samples, and the results from each processor can be pooled, leading to a significant reduction in the real-time performance of the algorithm. Also note that the sums over the interval $(0, T)$ required for the M-step can also be split into smaller independent time-segments which can be handled by different processors.

To minimize the bias in Model 1, one could again consider the computationally fairly expensive solution by using the particle filter approach

(Arulampalam et al. 2002). Particle filters are sequential Monte Carlo methods based on a point-mass representation of the relevant probability densities ($P(\vec{N}, t)$, in the current context), which can be applied to any state-space model (including models with nonlinear dynamics and non-Gaussian observations) and therefore generalize the EKS approach presented here. Particle filters, depending on the number of sample points chosen, can achieve any level of accuracy. This accuracy comes, however, at a significant computational cost: to obtain accurate results we need to sample from a potentially large set of particles.

The *unscented Kalman filter* (UKF) is another alternate approach (Wan and Merwe 2004); the idea is to use a minimal set of carefully chosen sample points that approximate the distribution. These sample points capture the true mean and covariance of the state of the system if the underlying dynamics are Gaussian and linear (as in our case), and when propagated through a nonlinearity, capture the posterior mean and the covariance to the second-order of the Taylor expansion of the nonlinearity. Incorporating the data observations in the UKF setting involves computing a sample average and a sample covariance, instead of the expansion approach taken by the EKS; like the particle filter, the UKF may be made more accurate (at some computational cost) by incorporating a larger number of sample points.

A final approach to overcome the bias issue in our estimates is to use the *Expectation Propagation* method (Minka 2001). Expectation propagation is a deterministic method that approximates the distributions using an iterative approach to incorporating the observations. It differs from EKS in that it may be applied to general (not only Gaussian) distributions. Expectation propagation is slightly more computationally intensive than EKS or UKF, but in general less intensive than particle filtering. In the future we plan to compare these different approaches in detail, and to apply these methods to real physiological data.

Acknowledgments

J.E. Kulkarni. would like to acknowledge the Swartz Foundation for a post-doctoral fellowship which supported this research; L. Paninski is supported by an NSF CAREER award, an Alfred P. Sloan Research Fellowship, and by NEI grant EY018003. We thank E.J. Chichilnisky, D. Nykamp, J. Pillow, M. Sahani, E. Simoncelli, B. Yu, and the members of the Center for Theoretical Neuroscience at Columbia University, for many helpful conversations.

Appendix: Deriving the forward-EKS equations

We derive the equations of the forward-EKS step, which computes the expectation and variance of the latent process $\vec{N}(t)$, conditioned on the observations made up to the time t. The filter is derived in an iterative manner, where we assume that we know the initial distribution of the hidden process. Over each subsequent time

interval we propagate this distribution through the dynamic equation, and update this transformed distribution based on the observed spiking of the neurons over that time interval.

- **Spike update**: Let there be a spike in the k-th neuron at time t. We will denote the time instant before and after spiking information is used by t^- and t^+, respectively. The forward Kalman filter is iterative and we assume that we know the distribution of the latent process at t^-, that is, $p(\vec{N}(t^-)|O_{0,t^-}) \sim \mathcal{N}(\vec{\mu}, \sigma^2)$. Then using the fact that the increments for a Poisson process are independent over disjoint intervals,

$$\begin{aligned} p(\vec{N}(t^+)|O_{0,t^-}, O_t^k = 1) &= \frac{1}{Z} p(\vec{N}(t^-)|O_{0,t^-}) p(O_t^k = 1|\vec{N}(t^-)) \\ &= \frac{1}{Z} \exp\left[-\frac{1}{2}(\vec{N} - \vec{\mu})' \sigma^{2^{-1}} (\vec{N} - \vec{\mu})\right] \exp[I_k + \vec{g}_k \vec{N}] \\ &= \frac{1}{Z} \exp\left[-\frac{1}{2}(\vec{N}' \sigma^{2^{-1}} \vec{N} - 2\vec{N}'(\sigma^{2^{-1}} \vec{\mu} + \vec{g}_k'))\right], \end{aligned}$$

  where Z is a normalizing factor. Therefore $p(\vec{N}(t^+)|O_{0,t^-}, O_t^k = 1) \sim \mathcal{N}(\vec{\mu} + \sigma^2 \vec{g}_k', \sigma^2)$.
- **For Non-spike updates**: Let there be no spike in any of the k neurons in $[t, t + \delta t]$, and let $p(\vec{N}(t + \delta t)|O_{0,t}) \sim \mathcal{N}(\vec{\mu}, \sigma^2)$, which is obtained from passing the updated distribution at time t through the linear dynamics Equations 17 and 18. We have,

$$p(\vec{N}(t + \delta t)|O_{0,t}, O_{t,t+\delta t} = \vec{0}) = \frac{1}{Z} p(\vec{N}(t + \delta t)|O_{0,t}) \prod_{k=1}^{K} p(O_{t,t+\delta t}^k = 0|\vec{N}(t + \delta t))$$

  Now,

$$\begin{aligned} p(O_t^k, t + \delta t = 0|\vec{N}(t + \delta t)) &= \exp[-\exp[I_k + \vec{g}_k \vec{N}]\delta t] \\ &= \exp\Big[-\exp[I_k + \vec{g}_k \vec{\mu}](1 + (\vec{N} - \vec{\mu})' \vec{g}_k \\ &\quad + \frac{1}{2}(\vec{N} - \vec{\mu})' \vec{g}_k' \vec{g}_k (\vec{N} - \vec{\mu})')\delta t\Big] + o(\delta t^2) \\ &= Z \exp\left[-\frac{W_k}{2}(\vec{N}' \vec{g}_k' \vec{g}_k \vec{N} - 2\vec{N}'(\vec{g}_k' \vec{g}_k \vec{\mu} - \vec{g}_k'))\right] \end{aligned}$$

  where I_k and $\vec{N}$ are evaluated at time $t + \delta t$, $W_k = -\exp(I_k + \vec{g}_k \vec{\mu})\delta t$, and Z is a normalizing factor.
  Therefore using results from the product of Gaussian distributions, we have

$$p\Big[\vec{N}(t + \delta t)|O_{0,t}, O_{t,t} + \delta t = \vec{0}\Big] \sim \mathcal{N}(\vec{\mu}_{\text{prod}}, \sigma_{\text{prod}}^2),$$

where

$$\sigma^2_{\text{prod}} = \left(\sigma^{2-1} + \sum_{k=1}^{K} = 1^K W_k \vec{g}'_k \vec{g}_k\right)^{-1}$$
$$= \sigma^2 - \sum_{k=1}^{K} \sigma^2 W_k \vec{g}'_k \vec{g}_k \sigma^2 + o(\delta t^2),$$
$$\vec{\mu}_{\text{prod}} = \vec{\mu} - \sigma^2 \left(\sum_{k=1}^{K} W_k \vec{g}'_k\right) + o(\delta t^2)$$

References

Abeles M. 1991. Corticonics. Neural Circuits of the Cerebral Cortex. New York: Cambridge University Press.

Andersen R, Musallam S, Pesaran B. 2004. Selecting the signals for a brain-machine interface. Curr Opin Neurobiol 14:720–726.

Arulampalam S, Maskell S, Gordon N, Clapp T. 2002. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. IEEE Trans Signal Process 50(2):174–188.

Boyd S, Vandenberghe L. 2004. Convex Optimization. New York: Oxford University Press.

Brillinger D. 1988. Maximum likelihood analysis of spike trains of interacting nerve cells. Biol Cyberkinet 59:189–200.

Brockwell A, Rojas A, Kass R. 2004. Recursive Bayesian decoding of motor cortical signals by particle filtering. J Neurophysiol 91:1899–1907.

Brody C. 1999. Correlations without synchrony. Neural Comput 11:1537–1551.

Brown E, Kass R, Mitra P. 2004. Multiple neural spike train data analysis: State-of-the-art and future challenges. Nature Neurosci 7:456–461.

Chornoboy E, Schramm L, Karr A. 1988. Maximum like- lihood identification of neural point process systems. Biol Cybernetics 59:265–275.

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

Cox D. 1955. Some statistical methods connected with series of events. J Royal Stat Soc, Ser B 17:129–164.

Dayan P, Abbott, L. 2001. Theoretical Neuroscience. Cambridge, Massachusetts: MIT Press.

de Jong P, MacKinnon M. 1988. Covariances for smoothed estimates in state space models. Biometrika 75:601–2.

Dempster A, Laird N, Rubin D. 1977. Maximum likelihood from incomplete data via the EM algorithm. J Royal Stat Soc, Ser B 39:1–38.

Dieci L, Eirola T. 1994. Positive definiteness in the numerical solution of riccati equations. Numer Math 67:303–13.

Donoghue J. 2002. Connecting cortex to machines: Recent advances in brain interfaces. Nat Neurosci 5:1085–1088.

Doucet A, de Freitas N, Gordon N, editors. 2001. Sequential Monte Carlo in Practice. New York: Springer.

Harvey A. 1991. Forecasting, structural time series models and the Kalman filter. New York: Cambridge University Press.

Huys Q, Zemel R, Natarajan R, Dayan P. 2007. Fast population coding. Neural Comput 19(2): 460–497.

Iyengar S. 2001. Advances in methodological and applied aspects of probability and statistics, chapter The analysis of multiple neural spike trains, pp. 507–524. New York: Taylor and Francis.

Jackson B. 2004. Including long-range dependence in integrate-and-fire models of the high interspike-interval variability of cortical neurons. Neural Comput 16:2125–2195.

Karatzas I, Shreve S. 1997. Brownian motion and stochastic Calculus. Springer.

Litke A, Bezayiff N, Chichilnisky E, Cunningham W, Dabrowski W, Grillo A, Grivich M, Grybos P, Hottowy P, Kachiguine S, Kalmar R, Mathieson K, Petrusca D, Rahman M, Sher A. 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 51(4): 1434–1440.

Loizou P. 1998. Mimicking the human ear: an introduction to cochlear implants. IEEE Signal Process Mag 15:101–130.

Martignon L, Deco G, Laskey K, Diamond M, Freiwald W, Vaadia E. 2000. Neural coding: Higher-order temporal patterns in the neuro-statistics of cell assemblies. Neural Comput 12:2621–2653.

Mendel J. 1995. Lessons in estimation theory for signal processing, communication, and control. Upper Saddle River, NJ: Prentice Hall.

Minka T. 2001. A family of algorithms for approximate bayesian inference. PhD thesis, MIT.

Moeller J, Syversveen A, Waagepetersen R. 1998. Log- Gaussian Cox processes. Scand J Stat 25:451–482.

Moeller J, Waagepetersen R. 2004. Statistical in ference and simulation for spatial point processes. Boca Raton, FL: Chapman Hall.

Nicolelis M, Dimitrov D, Carmena J, Crist R, Lehew G, Kralik J, Wise S. 2003. Chronic, multisite, multielectrode recordings in macaque monkeys. PNAS 100:11041–11046.

Nykamp D. 2005. Revealing pairwise coupling in linear-nonlinear net- works. SIAM J Appl Math 65:2005–2032.

Nykamp DQ. 2007. A mathematical framework for inferring connectivity in probabilistic neuronal networks. Math Biosci 205:204–251.

Okatan M, Wilson M, Brown E. 2005. Analyzing functional connectivity using a network likelihood model of ensemble neural spiking activity. Neural Comput 17:1927–1961.

Paninski L. 2004. Maximum likelihood estimation of cascade point-process neural encoding models. Network: Comput Neural Syst 15:243–262.

Paninski L. 2005. Log-concavity results on Gaussian process methods for supervised and unsupervised learning. Adv Neural Inform Process Syst 17:1025–1032.

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, Pillow J, Lewi J. 2008. Statistical models for neural encoding, decoding, optimal stimulus design. In Cisek P, Drew T, Kalaska J, editors, Computational Neuroscience Progress Brain Research. Burlington, MA: Elsevier.

Book RCO5 Robert C, Casella G. 2005. Monte Carlo Statistical Methods. Springer.

Pillow J, Paninski L, Shlens J, Simoncelli E, Chichilnisky E. 2005. Modeling multi-neuronal responses in primate retinal ganglion cells. Comp Sys Neur '05.

Plesser H, Gerstner W. 2000. Noise in integrate-and-fire neurons: From stochastic input to escape rates. Neural Comput 12:367–384.

Press W, Teukolsky S, Vetterling W, Flannery B. 1992. Numerical recipes in C. New York: Cambridge University Press.

Rabiner L. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 77:257–286.

Roweis S, Ghahramani, Z. 1999. A unifying review of linear gaussian models. Neural Comput 11(2): 305–345.

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

Salakhutdinov R, Roweis ST, Ghahramani Z. 2003. Optimization with EM and expectation-conjugate-gradient. Inter Conf Machine Learning 20:672–679.

Schnitzer M, Meister, M. 2003. Multineuronal firing patterns in the signal from eye to brain. Neuron 37:499–511.

Smith A, Brown, E. 2003. Estimating a state-space model from point process observations. Neural Comput 15:965–991.

Snyder D. 1972a. Filtering and detection for doubly stochastic poisson processes. IEEE Trans Inform Theory 18:91–102.

Snyder DL. 1972b. Smoothing for doubly stochastic poisson processes. IEEE Trans Inform Theory 18:558–562.

Snyder D, Miller M. 1991. Random point processes in time and space. New York: Springer-Verlag.

Stevens C, Zador A. 1996. When is an integrate-and-fire neuron like a Poisson neuron? NIPS 8:103–109.

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. J Neurophysiol 93:1074–1089.

Utikal K. 1997. A new method for detecting neural interconnectivity. Biol Cyberkinet 76:459–470.

Wan E, Merwe R. 2004. The unscented Kalman filter. In Haykin S, editor, Kalman Filtering and Neural Networks, pp. 131–158. Wiley-Intersceince.

Weiland J, Liu W, Humayun M. 2005. Retinal prosthesis. Ann Rev Biomed Eng 7:361–401.

Wu W, Black MJ, Mumford D, Gao Y, Bienenstock E, Donoghue JP. 2004. Modeling and decoding motor cortical activity using a switching Kalman filter. IEEE Trans Biomed Eng 51:933–942.

Wu W, Gao Y, Bienenstock E, Donoghue JP, Black MJ. 2005. Bayesian population coding of motor cortical activity using a Kalman filter. Neural Comput 18(1), Jan 2006:80–118.

Yu B, Afshar A, Santhanam G, Ryu S, Shenoy K, Sahani M. 2005. Extracting dynamical structure embedded in neural activity. NIPS.