Journal of Computational Neuroscience manuscript No.
(will be inserted by the editor)

Efficient computation of the maximum a posteriori path and parameter estimation in integrate-and-fire and more general state-space models

Shinsuke Koyama · Liam Paninski

Received: March 10, 2009/ Accepted: date

Abstract A number of important data analysis problems in neuroscience can be solved using state-space models. In this article, we describe fast methods for computing the exact maximum a posteriori (MAP) path of the hidden state variable in these models, given spike train observations. If the state transition density is log-concave and the observation model satisfies certain standard assumptions, then the optimization problem is strictly concave and can be solved rapidly with Newton-Raphson methods, because the Hessian of the loglikelihood is block tridiagonal. We can further exploit this block-tridiagonal structure to develop efficient parameter estimation methods for these models. We describe applications of this approach to neural decoding problems, with a focus on the classic integrate-and-fire model as a key example.

Keywords Tridiagonal Newton-Raphson method · Laplace approximation · State-space models · Point processes

1 Introduction

A number of important models in neuroscience may be described in "state-space" form with point-process observations: a hidden state variable evolves according to some continuous Markovian dynamics, and the rate of the observed spike trains is some function of this underlying hidden state. Examples include the integrate-and-fire model [1] and models used in a number of spike train decoding applications [2–5]; see [6] for a recent review.

It is of significant interest to compute the maximum *a posteriori* (MAP) path that the hidden state variable traversed on a given trial, given the observed spike trains;

S. Koyama
Department of Statistics and Center for the Neural Basis of Cognition, Carnegie Mellon University
Pittsburgh, PA, USA
E-mail: koyama@stat.cmu.edu

L. Paninski
Department of Statistics and Center for Theoretical Neuroscience, Columbia University
New York, NY, USA
E-mail: liam@stat.columbia.edu

this MAP path is essential both for decoding the dynamics of the hidden state given the spike train observations, and also for parameter estimation in this hidden Markov setting, and allows us to address a wide variety of biological problems such as inferring presynaptic inputs given postsynaptic voltage recording, detecting the location of a synapse given noisy voltage observations, and tracking nonstationary neural tuning properties. For reviews of these applications, see [6] and references therein.

The point-process filter algorithm introduced in [2] (adapted from earlier contributions in the statistical literature [7]) computes an approximation to this MAP path, but this approximation is rigorously accurate only in certain special limiting cases. More recently, [8] discussed methods for computing the MAP path exactly (without relying on the state-space framework), but without further assumptions these methods have computational complexity of $O(N^3)$, where N is the trial length, and are therefore inapplicable for long trials.

Here we develop $O(N)$ methods for computing the exact MAP path. In the case of linear Gaussian state space dynamics, the MAP path in continuous time satisfies a second-order nonlinear ordinary differential equation [9,10]. This equation may be solved numerically via standard relaxation or shooting methods [11]. More generally, if the dynamics are linear and driven by innovations with a log-concave density, and the point-process observations satisfy certain standard assumptions, then the optimization problem is strictly concave (guaranteeing a unique MAP path), and the Newton-Raphson algorithm may be applied; each Newton update here requires just $O(N)$ time, because the Hessian of the loglikelihood is block tridiagonal [7,12,13]. In the case of the integrate-and-fire neuron with a hard voltage threshold, barrier methods may be applied to solve the resulting constrained optimization problem [14].

In section 2, we develop the tridiagonal Newton-Raphson method for computing the MAP path of the hidden state in $O(N)$ time. Two other methods for state estimation, the point process filter/smoother and the expectation-propagation (EP) algorithm [15], are then described for comparison. In section 3, we show that the MAP path can be used for learning model parameters; maximizing the marginal likelihood function via Laplace approximation can again be done again in $O(N)$ time. We also discuss the relation of this method to approximate expectation-maximization (EM) algorithms for this model [16,3]. In section 4, we illustrate the application of these methods to the problem of predicting the subthreshold voltage trace from spike train data, and to inferring an unknown filter applied to the input current. We close with summary and discussion in section 5.

2 Inferring the MAP path

2.1 State-space model and the MAP path

We will first describe the class of models to be considered. Let $x(t)$ denote a (scalar- or vector-valued) diffusion process which is the solution of the following stochastic differential equation:

$$dx(t) = h(x(t))dt + \sigma dB(t), \tag{1}$$

where $B(t)$ is the standard Brownian motion. The state process $x(t)$ is not directly observable, but we observe a variable $y(t)$, which is related to $x(t)$, through a noisy process. This observation process is modeled by a probability density of $y(t)$ given the

state $x(t)$ at time t, denoted by $p(y(t)|x(t))$. This model class covers a wide range of state-space models, including linear and nonlinear state-space models [17] and dynamic generalized linear models [18,7]. In this article, we are especially interested in models with point process observations [19], in which the observation variable is taken to be a sequence of event times, $\{t_i\}$:

$$y(t) = \sum_i \delta(t - t_i), \tag{2}$$

where $\delta(t)$ denotes the Dirac delta function.

The stochastic integrate-and-fire (IF) model is an important special case of a state-space model with point process observations. It is the simplest model of neurons' spiking mechanism which consists of the membrane voltage dynamics and spiking threshold, and has been widely used for modeling neural dynamics as well as for data analytical purposes ([1] and references therein). The stochastic voltage process $x(t)$ of the simplest leaky IF model follows the linear stochastic dynamics,

$$dx(t) = -gx(t)dt + \sigma dB(t). \tag{3}$$

The probability with which a spike is observed in a small interval is written as

$$\Pr\{\text{a spike in } [t, t + dt)\} = f(x(t))dt, \tag{4}$$

where $f(x)$ is a nonnegative intensity function [19]. In the hard-threshold case, in which an observed spike is modeled as the first passage time of the voltage process $x(t)$ through the threshold voltage x_{th}, $f(x)$ is given by a step-function which takes the value zero for $x < x_{th}$ and jumps to ∞ at $x = x_{th}$. In the soft-threshold case, there is no explicit threshold, but $f(x)$ is given by a monotonically increasing continuous function.

Another example for multi-dimensional state-space models is a two-compartment leaky IF model which accounts for soma-dendric dynamics. Let $x_1(t)$ and $x_2(t)$ be the voltage at the soma and apical dendrite, respectively, and imagine that these variables follow the simple two-compartment stochastic dynamics

$$dx_1(t) = [-g_1 x_1(t) + a(x_2(t) - x_1(t))]dt + \sigma_1 dB_1(t) \tag{5}$$

$$dx_2(t) = [-g_2 x_2(t) + a(x_1(t) - x_2(t))]dt + \sigma_2 dB_2(t), \tag{6}$$

where $B_1(t)$ and $B_2(t)$ denote standard Brownian motions which are independent of each other. Here a is an intercompartmental coupling strength and g_i denotes the membrane leakiness of compartment i. Introducing the state vector $\boldsymbol{x}(t) = (x_1(t), x_2(t))^T$, with $(.)^T$ denoting the transpose, the state dynamics is written in the same form as (1),

$$d\boldsymbol{x}(t) = A\boldsymbol{x}(t)dt + \Sigma d\boldsymbol{B}(t), \tag{7}$$

where $\boldsymbol{B}(t)$ is the two-dimensional standard Brownian motion, and

$$A = \begin{pmatrix} -g_1 - a & a \\ a & -g_2 - a \end{pmatrix}, \quad \Sigma = \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix}. \tag{8}$$

The spike observation model is given by $\Pr\{\text{a spike in } [t, t + dt)\} = f(x_1(t))dt$. Of course, further multicompartmental generalizations are possible [20]; in addition, this state-space framework includes other types of IF neuron such as the resonate-and-fire

and the quadratic IF models [21], although our focus here will be on models in which the dynamics $h(x(t))$ are linear, for simplicity.

We wish to compute the MAP estimate of the state variable $x(t)$ given observations of the full spike train $\{y(t)|0 \leq t \leq T\}$. Let $p(\{x(t)\})$ be the probability distribution of $\{x(t)\} \equiv \{x(t)|0 \leq t \leq T\}$ which is induced by the Brownian $\{B(t)|0 \leq t \leq T\}$, and $p(\{y(t)\}|\{x(t)\})$ the conditional probability distribution of $\{y(t)|0 \leq t \leq T\}$ given $\{x(t)\}$. From Bayes' theorem, the log-posterior distribution of $\{x(t)\}$ is obtained as

$$\log p(\{x(t)\}|\{y(t)\}) = \log p(\{y(t)\}|\{x(t)\}) + \log p(\{x(t)\}) + \text{const.} \tag{9}$$

The MAP path of $\{x(t)\}$ is computed by maximizing the log-posterior distribution with respect to $\{x(t)\}$,

$$\{\hat{x}(t)\} = \arg\max_{\{x(t)\}} \big\{ \log p(\{y(t)\}|\{x(t)\}) + \log p(\{x(t)\}) \big\}. \tag{10}$$

In the following section, we describe efficient numerical algorithms to compute the MAP path. We will illustrate these methods with an application to the LIF model; the extension to vector-valued state-space models is straightforward.

2.2 Euler-Lagrange approach

We first review a previous continuous-time approach proposed for the computation of the MAP path. All the details are available in the previous papers [9,10,22,23]. The log-posterior distribution of $\{x(t)\}$ of the state-space model can be written as

$$\log p(\{y(t)\}|\{x(t)\}) + \log p(\{x(t)\}) = \int_0^T L(x, \dot{x}, t)dt + \text{const.} \tag{11}$$

where $L(x, \dot{x}, t)$ is defined as an appropriate *Lagrangian*, a functional of x, $\dot{x}$ and t. By applying the variational method, the MAP path $\hat{x}(t)$ is found to satisfy the Euler-Lagrange equation under an appropriate boundary condition,

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) - \frac{\partial L}{\partial x} = 0. \tag{12}$$

For example, if we apply the standard point-process loglikelihood formula [19], it is easy to see that the Lagrangian for the soft-threshold leaky IF model (3) is given by

$$L(x, \dot{x}) = \sum_i \delta(t - t_i) \log f(x) - f(x) - \frac{1}{2\sigma^2}(\dot{x} + gx)^2. \tag{13}$$

Thus the Euler-Lagrange equation is computed as

$$\ddot{x}(t) = g^2 x(t) + \sigma^2 \left[f'(x(t)) - \sum_i \delta(t - t_i) \frac{f'(x(t_i))}{f(x(t_i))} \right], \tag{14}$$

with boundary conditions: $\dot{x}(0) = -gx(0)$ and $\dot{x}(T) = -gx(T)$. The MAP path may now be obtained by solving this differential equation via standard numerical techniques, e.g. relaxation or shooting methods [11].

2.3 Computing the MAP path via block-tridiagonal Newton-Raphson optimization

The numerical method we consider here for computing the MAP path works by directly maximizing the posterior density of the state, rather than solving the Euler-Lagrange equation. As we will discuss further below, this optimization approach turns out to be exactly equivalent to the standard "relaxation" method for solving the Euler-Lagrange differential equation discussed above; however, we will see that there are a few advantages in treating the optimization problem directly (not least is that we can track the convergence to the optimal solution by directly monitoring our objective function).

We first switch from continuous to discrete time, for concreteness. In a discrete-time setting, we partition the observation interval into a discrete set of times, $\{t_i : 0 \leq t_0 < t_1 < \cdots < t_N \leq T\}$, where $t_i - t_{i-1} = \Delta$ for $i = 0, 1, \ldots, N$. All pertinent model components, such as the state and observation values, are then defined at these specified times. For convenience, we write x_i for $x(t_i)$ and y_i for $y(t_i)$. For the point process observation model, y_i is taken to be a binary variable of $\{0, 1\}$, letting 1 indicate that a spike has occurred within the corresponding time interval. The derivative, dx/dt, in (1) is replaced by $(x_{i+1} - x_i)/\Delta$. Applying the discretization to the original continuous-time state-space model yields the state transition density $p(y_i|x_i)$ and the observation probability $p(x_i|x_{i-1})$. Because of the Markov properties of the model, the log posterior density of $\{x_i\}$ is written as

$$\log p(\{x_i\}|\{y_i\}) = \log p(x_0) + \sum_{i=1}^{N} \log p(x_i|x_{i-1}) + \sum_{i=1}^{N} \log p(y_i|x_i) + \text{const.} \tag{15}$$

If the state transition density $p(x_i|x_{i-1})$ is log-concave in x_i and x_{i-1}, the initial state density $p(x_0)$ is log-concave, and the observation density $p(y_i|x_i)$ is also log-concave in x_i, it is then easy to see that the log-posterior distribution (15) is also log-concave in $\{x_i\}$ (since addition preserves concavity), and therefore computing the MAP path is a concave optimization problem. Furthermore, if $\log p(x_0)$, $\log p(x_i|x_{i-1})$ and $\log p(y_i|x_i)$ are all smooth functions of $\{x_i\}$, we may apply the standard Newton-Raphson method to solve this optimization problem [11].

To describe the algorithm, let $\boldsymbol{x} = (x_1, x_2, \ldots, x_N)^T$ and $\boldsymbol{y} = (y_1, y_2, \ldots, y_N)^T$, and x_0 be given as a boundary condition. Let

$$S(\boldsymbol{x}) = \sum_{i=1}^{N} \log p(y_i|x_i) + \sum_{i=1}^{N} \log p(x_i|x_{i-1}), \tag{16}$$

where we omit the initial distribution, $p(x_0)$, since we assume that it is given as a boundary condition. Now the key idea is that because of the Markov structure of the state-space model, the Hessian matrix, $J = \nabla\nabla_x S(\boldsymbol{x})$, becomes a tridiagonal matrix:

$$J = \begin{pmatrix} D_1 & B_{1,2} & 0 & \cdots & & 0 \\ B_{1,2}^T & D_2 & B_{2,3} & 0 & & \vdots \\ 0 & B_{2,3}^T & D_3 & B_{3,4} & \ddots & \\ \vdots & \ddots & & \ddots & \ddots & 0 \\ & & & B_{N-2,N-1}^T & D_{N-1} & B_{N-1,N} \\ 0 & \cdots & & 0 & B_{N-1,N}^T & D_N \end{pmatrix} \tag{17}$$

where

$$D_i = \frac{\partial^2}{\partial x_i^2} \log p(y_i|x_i) + \frac{\partial^2}{\partial x_i^2} \log p(x_i|x_{i-1}) + \frac{\partial^2}{\partial x_i^2} \log p(x_{i+1}|x_i) \tag{18}$$

(where we apply the appropriate boundary conditions for $i = 1$ and N), and

$$B_{i,i+1} = \frac{\partial^2}{\partial x_i \partial x_{i+1}} \log p(x_{i+1}|x_i). \tag{19}$$

(For d-dimensional state-space models, D_i and $B_{i,i+1}$ are $d \times d$-matrices, and hence J becomes a block-tridiagonal matrix with blocks of size d.) Thus, in the Newton step:

$$\hat{\boldsymbol{x}}^{(m+1)} = \hat{\boldsymbol{x}}^{(m)} - \delta, \tag{20}$$

where δ is obtained by solving the linear equation, $J\delta = \nabla_x S(\hat{\boldsymbol{x}}^{(m)})$, the search direction δ can be computed in $O(d^3 N)$ time (via the block-tridiagonal Thomas algorithm, which is a simplified form of Gaussian elimination) instead of the usual $O((dN)^3)$ required to solve a problem of size $\dim(\boldsymbol{x}) = dN$. In Matlab, a linear equation $Jx = b$ is efficiently solved using the notation $x = J\backslash b$; if J is sparse and block-tridiagonal, the $O(N)$ algorithm is used automatically.

We have introduced the above methods in the context of fully-observed spiking data $\{y_i\}$, but these techniques may be applied just as easily in the case that only a subset of the spike train is observed: we simply set the observation terms $\log p(y_i|x_i)$ to zero at times i where no data were observed (since in this case y_i is independent of x_i, so $p(y_i|x_i)$ is constant in x_i, and we may choose this constant to be one, for convenience). This preserves the concave and block-tridiagonal nature of the MAP optimization problem, so we may still obtain the solution in $O(T)$ time. For example, we can quickly compute the MAP path of $\boldsymbol{x}$ given an observation of a single spike. [24] and [25] emphasized that the spike-triggered average (STA) may be well-approximated by this MAP path in the low-noise limit; the Euler-Lagrange equations used to characterize the STA in these earlier papers may be considered the continuous-time limits of our block-tridiagonal solutions here (recall section 2.2).

Finally, we should emphasize that this block-tridiagonal form of the Hessian J in the state-space setting is well-known in the statistics literature [7,26,12,13]. Below we give some simple applications to the integrate-and-fire model, and discuss how to extend the method in the hard-threshold case, where the observation term $\log p(y_i|x_i)$ is discontinuous and therefore the Newton-Raphson method can not be applied directly.

2.3.1 Example 1: Leaky IF neuron with soft-threshold

For the soft-threshold IF neuron with the state equation (3) and intensity function $f(x)$, (16) is given by

$$S(\boldsymbol{x}) = l(\boldsymbol{x}) - \frac{1}{2}\boldsymbol{x}^T C^{-1} \boldsymbol{x} + \frac{N}{2} \log(2\pi\sigma^2 \Delta), \tag{21}$$

where

$$l(\boldsymbol{x}) = \sum_{i=1}^{N} \log p(y_i|x_i) = \sum_{i=1}^{N} \left[y_i \log f(x_i) - f(x_i)\Delta \right] \tag{22}$$

is the discretized point-process observation log-likelihood, and C is the prior covariance matrix of the state vector whose inverse is given by

$$C^{-1} = \frac{1}{\sigma^2 \Delta} \begin{pmatrix} 1+\alpha^2 & -\alpha & 0 & \cdots & 0 \\ -\alpha & 1+\alpha^2 & -\alpha & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & & -\alpha & 1+\alpha^2 & -\alpha \\ 0 & \cdots & 0 & -\alpha & 1 \end{pmatrix}, \tag{23}$$

where we define $\alpha = 1 - g\Delta$. Here, we assume that x_0 is given as a boundary condition and is excluded from the state vector $\boldsymbol{x}$. The Hessian matrix for this model is then obtained as $J = \nabla\nabla_x l(\boldsymbol{x}) - C^{-1}$, which is tridiagonal since $\nabla\nabla_x l(\boldsymbol{x})$ is a diagonal matrix. The log-likelihood $l(\boldsymbol{x})$ is concave, and therefore the log-posterior $S(\boldsymbol{x})$ is concave as well, when the function $f(.)$ is convex and log-concave [27]. Then our tridiagonal Newton-Raphson method can be applied to compute the MAP path in $O(N)$ time.

2.3.2 Example 2: Leaky IF neuron with hard-threshold

In the hard-threshold leaky IF model, a spike is evoked when $x(t) \geq x_{th}$, and $x(t)$ is reset to $x_{res} = 0$ immediately after spiking. Because of this hard threshold, computation of the MAP path should be treated as a constrained problem (specifically, a quadratic programming problem [10]). Through this example, we illustrate how to compute the MAP path under constraints by utilizing *barrier* (aka *interior-point*) methods [14].

Since the membrane potential is reset to x_{res} after each spike, inter-spike intervals are independent of each other, and can be treated independently [28]. Assuming that the voltage starts at time $i = 0$ and a spike is evoked at $i = N$, the MAP path is computed by maximizing (21) under the boundary conditions $x_0 = x_{res}$, and $x_N = x_{th}$, and the constraint $x_i < x_{th}$ for $i = 1, 2, \ldots, N-1$. To handle this constrained problem while exploiting the fast tridiagonal techniques, we can employ barrier methods. The idea is to replace the constrained problem

$$\hat{\boldsymbol{x}} = \arg \max_{\boldsymbol{x}: x_i \leq x_{th}} S(\boldsymbol{x}) \tag{24}$$

with a sequence of unconstrained problems

$$\hat{\boldsymbol{x}}_\epsilon = \arg\max_{\boldsymbol{x}} \left\{ S(\boldsymbol{x}) + \epsilon \sum_i \log(x_{th} - x_i) \right\}. \tag{25}$$

Clearly, $\hat{\boldsymbol{x}}_\epsilon$ satisfies the constraint $x_i \leq x_{th}$, since $\sum_i \log(x_{th} - x_i) \to -\infty$ as $x_i \to x_{th}$. Furthermore, if $\hat{\boldsymbol{x}}$ is unique, then $\hat{\boldsymbol{x}}_\epsilon$ converges to $\hat{\boldsymbol{x}}$ as $\epsilon \to 0$ [14]. Finally, the Hessian of the objective function $S(\boldsymbol{x}) + \epsilon \sum_i \log(x_{th} - x_i)$ retains the tridiagonal properties of the original objective $S(\boldsymbol{x})$; thus we can use our fast Newton iteration to obtain $\hat{\boldsymbol{x}}_\epsilon$ for any ϵ, and then sequentially decrease ϵ (in an outer loop) to obtain $\hat{\boldsymbol{x}}$.

We note that the barrier approach can be used more generally in multi-dimensional settings, whenever we want to enforce a convex constraint on x_i.

2.4 Point process filter and smoother

In section 4, we compare the performance of the MAP path algorithm described above against two other methods that have been applied previously. The first method is the point process filter and smoother [7,2,3], which is based on the Bayesian "forward-backward" recursion familiar from the theory of discrete-time hidden Markov models [29] or, in the case that both the transitions $p(x_{i+1}|x_i)$ and observations $p(y_i|x_i)$ are linear and Gaussian, the Kalman filter [30]. From the Markov properties of the state-space models, the conditional probability distribution of x_i given a sequence of observations up to time-step i, $y_{1:i} \equiv \{y_1, \ldots, y_i\}$ (the so-called "forward filter distribution"), may be expressed recursively as

$$p(x_i|y_{1:i}) = \frac{p(y_i|x_i)p(x_i|y_{1:i-1})}{\int p(y_i|x_i)p(x_i|y_{1:i-1})dx_i}, \tag{26}$$

where

$$p(x_i|y_{1:i-1}) = \int p(x_i|x_{i-1})p(x_{i-1}|y_{1:i-1})dx_{i-1}, \tag{27}$$

is the one-step "predictive" distribution. Starting with an initial distribution $p(x_0)$, the forward filter distribution $p(x_i|y_{1:i})$ and the predictive distribution $p(x_i|y_{1:i-1})$ for $i = 1, 2, \ldots, N$ can be computed recursively by applying (26) and (27). Note that this recursion computes the conditional distribution of x_i given the observations only up to the current time-step i. Once the filtered and predictive distributions in a whole time interval $(0, N)$ are obtained, the posterior (smoothed) distribution of x_i given a whole observation $y_{1:N}$ can be computed by recursing backwards:

$$p(x_i|y_{1:N}) = p(x_i|y_{1:i}) \int \frac{p(x_{i+1}|y_{1:N})p(x_{i+1}|x_i)}{p(x_{i+1}|y_{1:i})} dx_{i+1}. \tag{28}$$

for $i = N-1, N-2, \ldots, 1$.

Although the recursive equations (26), (27) and (28) can be solved by the well-known Kalman filter for linear Gaussian state-space models, they are not analytically tractable in general; in particular, in our case of point process observations, these recursions must be solved approximately. The point process filter and smoother approximates the filtered distribution (26) as a Gaussian, which provides a simple algorithm that is computationally tractable. The algorithm is given in detail in Appendix A; see also [7,2,5].

2.5 Expectation propagation algorithm

The second inference method to be compared is the expectation propagation (EP) algortihm [15]. We briefly summarize the application of the EP algorithm to state-space models, as previously discussed in [31–33]. The basic idea is to approximate the full joint distribution $p(\boldsymbol{x}|\boldsymbol{y})$ as a weighted Gaussian:

$$p(\boldsymbol{x}, \boldsymbol{y}) = w\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, C), \tag{29}$$

where $\mathcal{N}(x; \mu, C)$ denotes a Gaussian density in x with mean μ and C. Of course, as we have discussed above, this Gaussian approximation is exact in the case of the Kalman filter; unfortunately, more generally, the integrals

$$w = \int p(\boldsymbol{x}, \boldsymbol{y})d\boldsymbol{x}, \tag{30}$$

$$\mu_i = \int p(\boldsymbol{x}|\boldsymbol{y}) x_i d\boldsymbol{x}, \tag{31}$$

and

$$C_{ij} = \int p(\boldsymbol{x}|\boldsymbol{y})(x_i - \mu_i)^T (x_j - \mu_j) d\boldsymbol{x} \tag{32}$$

that define w, μ, and C are not directly tractable when the observations $\{y_i\}$ are given by a point-process. The EP algorithm approximates these quantities iteratively, by incorporating the non-Gaussian terms $p(y_i|x_i)$ one-by-one into our Gaussian approximation; each such iteration requires a one-dimensional numerical integral and a rank-one update of the approximate covariance matrix C. See [15] for details in the general case.

In the state-space case, we can cast the EP algorithm in terms of the one- and two-slice marginal posterior state distributions $p(x_i|y_{1:N})$ and $p(x_{i-1}, x_i|y_{1:N})$. These distributions can be expressed in terms of forward and backward "messages," $\alpha_i(x_i)$, $\beta_i(x_i)$, as

$$p(x_i|y_{1:N}) = \alpha_i(x_i)\beta_i(x_i), \tag{33}$$

and

$$p(x_{i-1}, x_i|y_{1:N}) = \frac{1}{c_i}\alpha_{i-1}(x_{i-1})p(y_i|x_i)p(x_i|x_{i-1})\beta_i(x_i), \tag{34}$$

where

$$c_i = p(y_i|y_{1:i-1}), \tag{35}$$

$$\alpha_i(x_i) = \frac{1}{c_i}p(y_i|x_i)\int p(x_i|x_{i-1})\alpha_{i-1}(x_{i-1})dx_{i-1}, \tag{36}$$

and

$$\beta_{i-1}(x_{i-1}) = \frac{1}{c_i}\int p(y_i|x_i)p(x_i|x_{i-1})\beta_i(x_i)dx_i. \tag{37}$$

The forward (36) and the backward (37) recursions for the messages correspond to the standard forward-backward recursions in the discrete-time, discrete-space hidden Markov model [29], and therefore are closely related to the recursive Bayesian equations (26)-(28) introduced above; in particular, eqs. (36-37) are equivalent to the Kalman filter and smoother when the state-space model is linear Gaussian.

To compute the messages $\{\alpha_i\}$ and $\{\beta_i\}$ for general nonlinear and non-Gaussian cases, these messages are typically approximated by an unnormalized Gaussian (or more generally an exponential family density). These approximate messages are iteratively updated by matching the expected sufficient statistics of the marginal posterior (33) with those of the two-slice marginal posterior (34). Although the convergence is not guaranteed for EP, if it converges it ends up in a minimum of the Bethe free energy [31]; in our experience the EP algorithm always converges here.

The updates are performed sequentially via multiple forward-backward passes. During the forward pass, the α_t are updated while the β_t are fixed,

$$\begin{aligned}\alpha_i(x_i) &= \int p(x_{i-1}, x_i|y_{1:N})dx_{i-1} \Big/ \beta_i(x_i) \\ &\approx q_i(x_i)/\beta_i(x_i),\end{aligned} \tag{38}$$

where $q_i(x_i)$ is a Gaussian approximation of $\int p(x_{i-1}, x_i|y_{1:N})dx_{i-1}$. The backward pass proceeds similarly, where the β_i are updated while the α_i remain fixed,

$$\begin{aligned}\beta_{i-1}(x_{i-1}) &= \int p(x_{i-1}, x_i|y_{1:N})dx_i \Big/ \alpha_{i-1}(x_{i-1}) \\ &\approx q_{i-1}(x_{i-1})/\alpha_{i-1}(x_{i-1}).\end{aligned} \tag{39}$$

The Gaussian approximation of the marginal density $q_s(x_s)$ ($s = i$ for the forward path, and $s = i - 1$ for the backward path) is obtained by matching the first two moments (i.e., the mean and variance) of $\int p(x_{i-1}, x_i|y_{1:N})d\backslash x_s$ (where $\backslash x_s$ is the other of the two neighbours, i.e., x_i or x_{i-1} respectively). Although the first two moments cannot be computed analytically in general, they can be approximated using standard one-dimensional numerical integration methods (See Appendix B). In the case of a multidimensional state-space x_i, the compuation of $q_i(x_i)$ requires a multidimensional integral, which in turn can be approximated by the standard EP algorithm, or alternatively it is possible to incorporate the non-Gaussian data one element of x_i at a time, using partial Kalman sweeps; again, see [31–33] for details.

One forward and backward sweep of the EP algorithm costs $O(N)$ time, and the number of iterations for convergence does not scale with the simulation interval N, in practice. Thus, the total computational time of the EP algorithm remains $O(N)$, as desired, although in practice EP is significantly slower than the other two approaches described above (due to the repeated numerical integration calls).

3 Parameter estimation

3.1 Laplace approximation

While we have focused so far on inferring the state path under the assumption that the correct model is known, in practice the state-space model includes unknown parameters θ, and one would like to estimate these parameters from observations. In principle, one could estimate the parameters by maximizing the marginal likelihood [12,34],

$$\log p(\boldsymbol{y};\theta) = \log \int p(\boldsymbol{y}|\boldsymbol{x};\theta)p(\boldsymbol{x};\theta)d\boldsymbol{x} = \log \int \prod_{i=1}^{N} p(x_i|x_{i-1};\theta)p(y_i|x_i;\theta)d\boldsymbol{x}. \quad (40)$$

However, in the non-Kalman case, it is intractable to compute the marginal likelihood exactly, since (40) involves a high-dimensional non-Gaussian integral. Here we show that the Laplace approximation centered by the MAP path can provide an efficient ($O(N)$) marginal likelihood computation, along with the gradients of the marginal likelihood.

We will illustrate our method in section 4.3 with the filtered-input IF model with soft-threshold,

$$dx(t) = [-gx(t) + \sum_i a_i u_i(t)]dt + \sigma dB(t), \quad (41)$$

where $\{u_i(t)\}$ are given inputs, and we wish to estimate the weights $\theta = \{a_i\}$; thus the reader may find it helpful to keep this model in mind here. Note that the log marginal likelihood is concave in $\{a_i\}$ under the usual convex and log-concave conditions on the soft-threshold nonlinearity (see [35] for details); this encourages the use of the Laplace approximation below.

To begin, recall the terminology introduced in section 2.3. The marginal likelihood function can be approximated as

$$\begin{aligned} p(\boldsymbol{y};\theta) &= \int \exp[S(\boldsymbol{x};\theta)]d\boldsymbol{x} \\ &\approx \exp[S(\hat{\boldsymbol{x}};\theta)] \int \exp\left[\frac{1}{2}(\boldsymbol{x}-\hat{\boldsymbol{x}})^T J(\boldsymbol{x}-\hat{\boldsymbol{x}})\right] d\boldsymbol{x} \\ &= (2\pi)^{N/2} \exp[S(\hat{\boldsymbol{x}};\theta)]|J|^{-1/2}, \end{aligned} \tag{42}$$

where we have introduced the Taylor expansion of $S(\boldsymbol{x};\theta)$ at the MAP path $\hat{\boldsymbol{x}}$ up to the second-order term (assuming that $S(\boldsymbol{x};\theta)$ is sufficiently smooth at $\hat{\boldsymbol{x}}$),

$$S(\boldsymbol{x};\theta) \approx S(\hat{\boldsymbol{x}};\theta) + \frac{1}{2}(\boldsymbol{x}-\hat{\boldsymbol{x}})^T J(\boldsymbol{x}-\hat{\boldsymbol{x}}), \tag{43}$$

where J is the Hessian evaluated at the MAP path $\hat{\boldsymbol{x}}$,

$$J = \nabla\nabla_x l(\hat{\boldsymbol{x}},\theta) - C(\theta)^{-1}, \tag{44}$$

with $l(.)$ denoting the log-likelihood $\log p(\boldsymbol{y}|\boldsymbol{x},\theta)$ and $C(\theta)$ the prior covariance of $\boldsymbol{x}$. The first term in the Taylor expansion is zero here, by the definition of $\hat{\boldsymbol{x}}$. (Note that we will suppress this dependence on θ below to keep the notation somewhat more legible.)

Taking logarithms, the marginal loglikelihood is thus approximated as

$$\begin{aligned} \log p(\boldsymbol{y};\theta) &\approx S(\hat{\boldsymbol{x}};\theta) - \frac{1}{2}\log|J| + \text{const.} \\ &= l(\hat{\boldsymbol{x}},\theta) - \frac{1}{2}\hat{\boldsymbol{x}}^T C^{-1}\hat{\boldsymbol{x}} - \frac{1}{2}\log|J| + \text{const.} \end{aligned} \tag{45}$$

To derive the partial derivative of the negative log likelihood function, note that $\hat{\boldsymbol{x}}$ is implicitly a function of θ. Thus,

$$\begin{aligned} \frac{\partial \log p(\boldsymbol{y};\theta)}{\partial\theta} &\approx [\nabla_x S(\hat{\boldsymbol{x}};\theta)]^T \frac{\partial\hat{\boldsymbol{x}}}{\partial\theta} + \frac{\partial S(\hat{\boldsymbol{x}};\theta)}{\partial\theta} - \frac{1}{2}\text{tr}\left[J^{-1}\frac{dJ}{d\theta}\right] \\ &= [\nabla_x l(\hat{\boldsymbol{x}},\theta)]^T \frac{\partial\hat{\boldsymbol{x}}}{\partial\theta} - \hat{\boldsymbol{x}}^T C^{-1}\frac{\partial\hat{\boldsymbol{x}}}{\partial\theta} - \frac{1}{2}\hat{\boldsymbol{x}}^T \frac{\partial C^{-1}}{\partial\theta}\hat{\boldsymbol{x}} - \frac{1}{2}\text{tr}\left[J^{-1}\frac{dJ}{d\theta}\right], \end{aligned} \tag{46}$$

where $\partial\hat{\boldsymbol{x}}/\partial\theta$ is obtained by differentiating the equation $\nabla_x S(\hat{\boldsymbol{x}};\theta) = 0$:

$$\frac{\partial}{\partial\theta}[\nabla_x S(\hat{\boldsymbol{x}};\theta)] = \nabla\nabla_x l(\hat{\boldsymbol{x}},\theta)\frac{\partial\hat{\boldsymbol{x}}}{\partial\theta} - \frac{\partial C^{-1}}{\partial\theta}\hat{\boldsymbol{x}} - C^{-1}\frac{\partial\hat{\boldsymbol{x}}}{\partial\theta} = 0. \tag{47}$$

Using (44), one obtains

$$\frac{\partial\hat{\boldsymbol{x}}}{\partial\theta} = J^{-1}\frac{\partial C^{-1}}{\partial\theta}\hat{\boldsymbol{x}}. \tag{48}$$

Note also that $J = J(\hat{\boldsymbol{x}},\theta)$ depends on θ directly and through $\hat{\boldsymbol{x}}$ implicitly; thus the derivative of J in the trace in (46) is expressed as

$$\begin{aligned} \frac{dJ}{d\theta} &= (\nabla_x J)\frac{\partial\hat{\boldsymbol{x}}}{\partial\theta} + \frac{\partial J}{\partial\theta} \\ &= (\nabla_x J)\frac{\partial\hat{\boldsymbol{x}}}{\partial\theta} + \frac{\partial[\nabla\nabla_x l(\hat{\boldsymbol{x}},\theta)]}{\partial\theta} - \frac{\partial C^{-1}}{\partial\theta}, \end{aligned} \tag{49}$$

where $\nabla_x J$ here abbreviates $\nabla_x J(x)|_{x=\hat{\boldsymbol{x}}}$. For example, in the soft-threshold LIF case discussed in section 2.3.1, $\nabla_x J$ may be represented as a diagonal matrix involving the third derivative of the link function $f(.)$, since each element of $\hat{\boldsymbol{x}}$ only effects the single corresponding diagonal element of J, which in turn includes terms depending on the second derivative of $f(.)$.

Three terms should be checked for computational cost: $\log|J|$ in (45), $\partial\hat{\boldsymbol{x}}/\partial\theta$ in (48), and the trace of the matrix in (46). Here the key in reducing computational cost is, again, that the Hessian J is a block-tridiagonal matrix. First, we can compute $\log|J|$ by

$$\frac{1}{2}\log|J| = \sum_i \log\left[J^{1/2}\right]_{ii}, \tag{50}$$

where $J^{1/2}$ is chosen to be the (triangular) Cholesky decomposition of J; the Cholesky decomposition may be computed here in $O(N)$ time using the standard banded or block-tridiagonal algorithm. (Note that the apparently simpler Matlab command "log(det(J))" for computing $\log|J|$ is very prone to numerical overflow errors, due to the large dimensionality of J, and is therefore not recommended.) Second, the linear equation (48) can be solved in $O(N)$ time, due to the block-tridiagonal nature of J, as discussed above. Finally, to compute the diagonal elements in the trace in (46), only a band of width $3d$ (where $d = 1$ for the leaky IF example) about the diagonal of J^{-1} is necessary because $dJ/d\theta$ is (block-)tridiagonal, and thus the total cost for computing (46) remains $O(N)$; see, e.g., [36–38] for details. Thus, to summarize, the log likelihood function and its derivative are computed in $O(N)$ ($O(Nd^3)$ for d-dimensional state-space models), and therefore parameter estimation via marginal likelihood optimization can be performed quite tractably via by standard gradient-based algorithms.

We may go a bit further if we note that, in the limit of small noise $\sigma^2 \ll 1$, the first two terms in (45) (which grow roughly linearly in σ^2) dominate the third term (which grows roughly logarithmically in σ^2). To maximize the first two terms with respect to θ together,

$$\hat{\theta} = \arg\max_\theta \left[l(\hat{\boldsymbol{x}}) - \frac{1}{2}\hat{\boldsymbol{x}}^T C^{-1}\hat{\boldsymbol{x}}\right] = \arg\max_\theta \max_{\boldsymbol{x}} \left[l(\boldsymbol{x}) - \frac{1}{2}\boldsymbol{x}^T C^{-1}\boldsymbol{x}\right], \tag{51}$$

we just need to optimize $S(\boldsymbol{x};\theta)$ jointly in $(\boldsymbol{x},\theta)$. (See [39] for a somewhat related physics-based optimization approach for fitting dynamical systems models.) In many cases $S(\boldsymbol{x};\theta)$ is a jointly concave function of $(\boldsymbol{x},\theta)$; since $S(\boldsymbol{x};\theta)$ and its derivatives may be computed quite easily, this significantly simplifies the computation of $\hat{\theta}$. The required joint optimization in $(\boldsymbol{x},\theta)$ is tractable due to the special structure of the Hessian matrix here. If we order the parameter vector as $\{\theta,\boldsymbol{x}\}$, the Hessian can be written in block form:

$$J = \begin{pmatrix} J_{\theta\theta} & J_{\theta\boldsymbol{x}}^T \\ J_{\theta\boldsymbol{x}} & J_{\boldsymbol{x}\boldsymbol{x}} \end{pmatrix}, \tag{52}$$

where $J_{\boldsymbol{x}\boldsymbol{x}}$ is block-tridiagonal. We cannot directly apply our tridiagonal Newton methods to obtain $\hat{\theta}$, since J itself is not tridiagonal. However, since $\dim(\theta) \ll N$, we can take the Schur complement,

$$J^{-1} = \begin{pmatrix} I & 0 \\ -J_{\boldsymbol{x}\boldsymbol{x}}^{-1}J_{\theta\boldsymbol{x}} & I \end{pmatrix} \begin{pmatrix} (J_{\theta\theta} - J_{\theta\boldsymbol{x}}^T J_{\boldsymbol{x}\boldsymbol{x}}^{-1} J_{\theta\boldsymbol{x}})^{-1} & 0 \\ 0 & J_{\boldsymbol{x}\boldsymbol{x}}^{-1} \end{pmatrix} \begin{pmatrix} I & -J_{\boldsymbol{x}\boldsymbol{x}}^{-1}J_{\theta\boldsymbol{x}}^T \\ 0 & I \end{pmatrix}, \tag{53}$$

to establish that computing the Newton step here for simultaneously obtaining $\{\hat{\theta}, \hat{\boldsymbol{x}}\}$ requires just $O(N)$ time. Since this optimization can be done much more quickly than the full conjugate gradient ascent on the full objective function (45), this makes for a very good initialization strategy.

We should also note, in passing, that it is possible to exploit these Schur complement ideas to develop $O(N)$ EP algorithms for approximating the marginal likelihood and the posterior mean of θ given the observed data $\{y(t)\}$, but we will not pursue this strategy further here.

Finally, it is worth establishing some connections between this direct optimization strategy for estimating θ and the so-called augmented filter algorithm discussed by [5]. In the augmented filter algorithm, the unknown parameters θ are incorporated into the state space as $u_i = (x_i, \theta)^T$. We then simply run the forward filter (i.e., the point process filter) to obtain $p(u_N|y_{1:N})$, from which we can easily extract the Gaussian likelihood $p(\theta|y_{1:N})$ by marginalizing $p(u_N|y_{1:N})$ with respect to x_N. In the direct optimization approach, we set up a joint optimization on $(\boldsymbol{x}, \theta)$ and use the Schur decomposition to perform this optimization efficiently. Our direct optimization approach has three major advantages here: first, our inference is based on the exact MAP path, instead of the approximate forward filter path computed by the augmented filter algorithm. Second, the augmented filter algorithm assumes that the posterior distribution of θ is approximately Gaussian, which may be a crude approximation in many cases, particularly when $S(\boldsymbol{x}, \theta)$ is not jointly strictly concave in $(\boldsymbol{x}, \theta)$. Finally, the Schur-based approach is faster in the case of a high-dimensional parameter vector θ, since the Schur complement requires just a single inversion of a $\dim(\theta) \times \dim(\theta)$ matrix, whereas the augmented filter algorithm requires that we compute the sufficient statistics of the $(d+\dim(\theta))$-dimensional vector u_i (and therefore invert a $(d+\dim(\theta)) \times (d + \dim(\theta))$ matrix) on each time step.

3.2 EM algorithm

An alternative way to approximately maximize the marginal likelihood function (40) is to use the expectation-maximization (EM) algorithm [16,3]. The procedure of the EM algorithm consists of the following two steps:

E-step Given the current estimate θ^{old} of θ, compute the posterior expectation of the log joint probability density,

$$Q(\theta|\theta^{old}) = E_x \left[\log p(\boldsymbol{x}, \boldsymbol{y}|\theta) \mid \boldsymbol{y}, \theta^{old} \right]. \tag{54}$$

M-step Update the estimate θ^{new} by

$$\theta^{new} = \arg\max_{\theta} Q(\theta|\theta^{old}).$$

These two steps are iterated until the estimation converges. When the complete data distribution is of an exponential family, the E-step consists of finding the expected values of certain sufficient statistics of the complete data. In our specific state-space setting, these expectations (specifically, the first and second moments of the pairwise conditional distributions $p(x_i, x_{i+1}|\boldsymbol{y})$) can be computed approximately, in $O(N)$ time, by the point process smoother, the EP algorithm, or by the direct Laplace approximation methods discussed above. For example, we may use the MAP path to approximate

the conditional expectations $E(x_i|\boldsymbol{y})$, and use the diagonal elements of the inverse Hessian matrix J to approximate the second moments $Var(x_i|\boldsymbol{y})$ (recall that these diagonal elements of the inverse Hessian may be obtained in $O(N)$ time, because the full matrix inverse is not required [36–38]).

Although the EM algorithm is fairly standard, many authors have reported that convergence tends to be unnecessarily slow ([40,41] and references therein). We have seen similar effects here; in practice, we have found that directly optimizing the marginal likelihood is much faster than iterating the EM algorithm to convergence. That said, it is worth noting that the two methods are quite closely related; in particular, it is well-known that the gradient of the marginal likelihood can be computed via the E-step as

$$\frac{\partial \log p(\boldsymbol{y};\theta)}{\partial \theta} = \frac{\partial Q(\theta|\tilde{\theta})}{\partial \theta}\bigg|_{\tilde{\theta}=\theta}. \tag{55}$$

Thus, the computation of the gradient of log marginal likelihood function can in general be carried out via the E-step of the EM algorithm [16,40,41].

In our case, the gradient of the log-likelihood function (55) can be connected explicitly to the Laplace approximation (46). Expanding the log-joint probability distribution in (54) around the MAP path $\hat{\boldsymbol{x}}$, $Q(\theta|\tilde{\theta})$ is approximated as

$$\begin{aligned} Q(\theta|\tilde{\theta}) &= \int p(\boldsymbol{x}|\boldsymbol{y};\tilde{\theta}) S(\boldsymbol{x};\theta) d\boldsymbol{x} \\ &\approx \int p(\boldsymbol{x}|\boldsymbol{y};\tilde{\theta}) \Big[S(\hat{\boldsymbol{x}};\theta) + \frac{1}{2}(\boldsymbol{x}-\hat{\boldsymbol{x}})^T J(\boldsymbol{x}-\hat{\boldsymbol{x}}) \Big] d\boldsymbol{x} \\ &\approx S(\hat{\boldsymbol{x}};\theta) + \frac{1}{2}\int \mathcal{N}(\boldsymbol{x};\hat{\boldsymbol{x}}(\tilde{\theta}), -J(\tilde{\theta})^{-1})(\boldsymbol{x}-\hat{\boldsymbol{x}})^T J(\boldsymbol{x}-\hat{\boldsymbol{x}}) d\boldsymbol{x}. \end{aligned} \tag{56}$$

In the last equation, the posterior distribution in the integral is replaced by its Laplace approximation. Taking the derivative of the second term in the right-hand-side of (56) with respect to θ and evaluating at $\tilde{\theta} = \theta$, we have

$$\begin{aligned} &\int \mathcal{N}(\boldsymbol{x};\hat{\boldsymbol{x}}(\tilde{\theta}), -J(\tilde{\theta})^{-1}) \Big[-2J(\boldsymbol{x}-\hat{\boldsymbol{x}}) \frac{\partial \hat{\boldsymbol{x}}}{\partial \theta} + (\boldsymbol{x}-\hat{\boldsymbol{x}})^T \frac{\partial J}{\partial \theta}(\boldsymbol{x}-\hat{\boldsymbol{x}}) \Big] d\boldsymbol{x} \bigg|_{\tilde{\theta}=\theta} \\ &\quad = \int \mathcal{N}(\boldsymbol{x};\hat{\boldsymbol{x}}, -J^{-1})(\boldsymbol{x}-\hat{\boldsymbol{x}})^T \frac{\partial J}{\partial \theta}(\boldsymbol{x}-\hat{\boldsymbol{x}}) d\boldsymbol{x} \\ &\quad = -\mathrm{tr}\Big[J^{-1} \frac{\partial J}{\partial \theta} \Big], \end{aligned} \tag{57}$$

which corresponds to the trace of the log-determinant term in (46), further clarifying the connection between the gradients of the log-likelihood computed by the Laplace approximation and the E-step. Eq. (57) also provides an interpretation for computing the derivative of the log-determinant $\log |J|$ in terms of the E-step.

4 Applications

4.1 Leaky IF neuron with soft-threshold

We applied the three methods (the MAP method, the point process smoother and the EP algorithm) to simulated spike trains generated from the soft-threshold leaky IF

neuron. The membrane dynamics was given by

$$dx(t) = [-50x(t) + I(t)]dt + \sigma dB(t). \tag{58}$$

The input $I(t)$ was generated by sampling from an Ornstein-Uhlenbeck process whose mean and covariance were $E(I(t)) = 60$ and $Cov(t, t') = 400e^{-10|t-t'|}$, respectively. We assumed the input was known for simplicity here; in section 4.3 we discuss the somewhat more realistic case in which we must infer an unknown pre-filter to accurately estimate the input $I(t)$. The state $x(t)$ is reset to $x_{res} = 0$ immediately after spiking, which can be incorporated by imposing the boundary condition of $x(t_i^+) = x_{res}$ after each spike. The soft-threshold function was set to $f(x) = e^x$. We first simulated the model in $N = 1000$ time-steps to generate a spike train, and the state was reconstructed from the spike train by the three methods (Fig. 1). In each case, the true input current $I(t)$ was considered known, and was included in our inference (via the transition density $p(x_{i+1}|x_i)$).

Fig. 2 shows the mean integrated squared error (MISE) between the true state $\boldsymbol{x}$ and the inferred one as a function of σ. The EP approximation gives the best estimation, followed by the MAP path and the point process smoother. When σ is small, the three approximations are very similar since the posterior is close to Gaussian and the posterior mode and mean are also close to each other (Fig. 1(a)). When σ is large, however, the EP approximation is better than the others, because the actual posterior distribution is far from Gaussian and the posterior mode is not close to the posterior mean (Fig. 1(b)). In such a case, the posterior mean can of course provide a better estimation under square error loss than the MAP estimate.

All three methods approximate the posterior distribution $p(x_i|\boldsymbol{y})$ as Gaussian. Under this Gaussian approximation, the variable

$$z_i = \frac{x_i - m_i}{e_i} \tag{59}$$

should be standard Gaussian, where x_i is the true state variable at time i, m_i is the conditional mean we have inferred, and e_i is the inferred conditional standard deviation. We tested the quality of this approximation by constructing Q-Q plots of z_i against the standard Gaussian quantiles. Fig. 3 depicts the results for $\sigma = 40$, in which the posterior mean is a better estimate than the MAP path. The Q-Q plot for the EP method is closer to the 45-degree reference line than the other methods. The Q-Q plots for the Laplace approximation centered by the MAP path and for the point process smoother, on the other hand, deviate from the 45-degree reference line, again indicating that the Laplace approximation of the posterior centered by the MAP path is biased in this high-σ case.

4.2 Leaky IF neuron with hard-threshold

Next, we applied the methods to simulated spike trains generated from the leaky IF neuron (58) with hard-threshold. The threshold and resting potential were taken to be $x_{th} = 1$ and $x_{res} = 0$, and the known input $I(t)$ was given by the Ornstein-Uhlenbeck process whose mean and covariance were $E(I(t)) = 5$ and $Cov(t, t') = 250e^{-20|t-t'|}$, respectively. The model was approximated in discrete-time with bin-size $\Delta = 0.001$, in the same way as the soft-threshold case. We first simulated the model in $N = 1000$

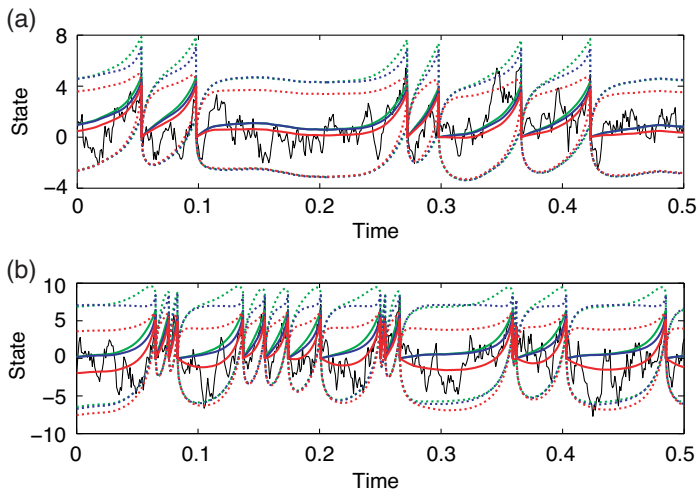

Fig. 1 Estimating the voltage path $x(t)$ in a soft-threshold LIF model. Trajectory of inferred state (colored solid lines) and approximate 95% Gaussian confidence interval (dotted lines). The black line represents the true trajectory. Blue: MAP path, green: the point process smoother, red: The EP algorithm. (a) results for $\sigma = 20$ and (b) for $\sigma = 40$. Note that the differences between the three methods become larger as σ grows.

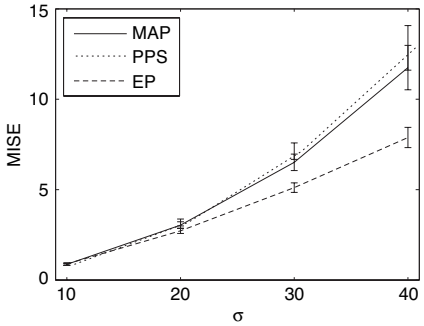

Fig. 2 The MISE between the actual and inferred states in a soft-threshold LIF model, as a function of σ. MAP, PPS and EP represent the MAP path, the point process smoother and the EP algorithm, respectively. The mean and standard error were calculated with 10 repetitions. The posterior expectation (as approximated by the EP algorithm) gives a better approximation than the MAP when σ is large; conversely, as $\sigma \to 0$, the Laplace approximation is accurate, and the three methods give similar results.

time-steps to generate a spike train, and then the state was reconstructed from the spike train. Results are shown in Fig. 4-6. Due to the hard threshold, the posterior distribution is farther from Gaussian than in the soft-threshold case for large value of σ (see further discussion in [25]), and thus the superiority of the EP algorithm against the MAP method becomes clearer.

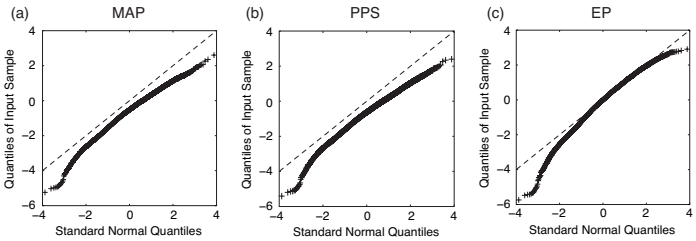

Fig. 3 Q-Q plots for the soft threshold LIF model with $\sigma = 40$. The Q-Q plots for the MAP method and the point process smoother deviate from the 45-degree reference line, as compared with the EP approximation, indicating a significant bias in the Laplace approximation at this value of σ.

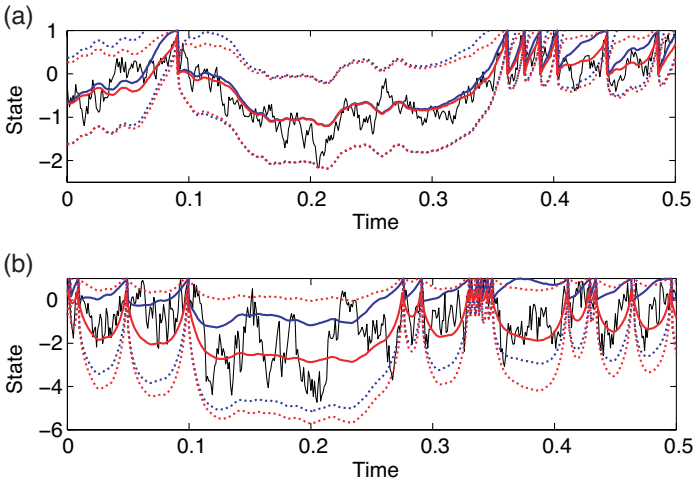

Fig. 4 Estimating the voltage path $x(t)$ in a hard-threshold LIF model. Conventions as in Fig. 1. (a) result for $\sigma = 5$, (b) result for $\sigma = 20$. Note that the conditional mean (EP path) sags well below the MAP path, particularly for large σ, as discussed in [25].

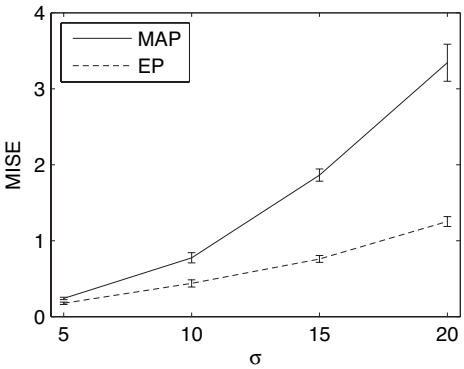

Fig. 5 The MISE between the actual and inferred states as a function of σ; results are similar to those shown in Fig. 2.

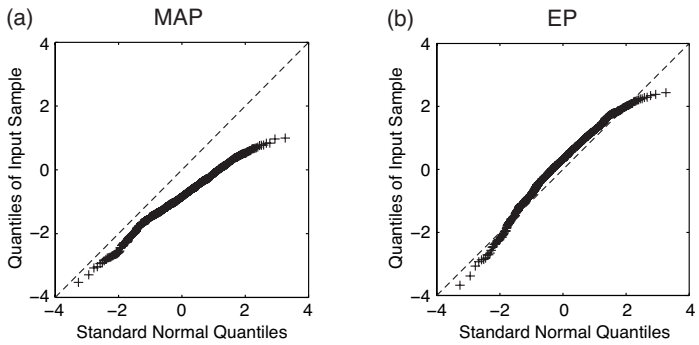

Fig. 6 Q-Q plots for the hard threshold LIF model with $\sigma = 20$. The Q-Q plot for the MAP method is deviated from the 45-degree reference line compared with the other two methods. Again, we see similar results as in the soft-threshold case (Fig. 3).

4.3 Inferring the input filter

The last example is to infer an unknown input filter, given a known input $I(t)$ and spike response. We took the intensity function in this simulation to be

$$f(x(t)) = \exp\left[x(t) + \sum_i a_i I(t-i)\right], \tag{60}$$

where a_i represent the weights of an unknown temporal filter. We let the state variable x follow the simple linear dynamics (3) here. In the simulation study, we inferred the filter k by two methods: the full marginal likelihood solution (45), and the simpler, faster initialization (i.e., dropping the log-determinant term (51)). First, we simulated spike trains with (3) and (60) with the parameter values $g = 50$ and $\sigma = 1$. The input $I(t)$ was taken to be a Gaussian white noise with unit variance. We took the original filter to be an α-function:

$$a_t = 0.05\frac{t}{\tau}\exp\Big(-\frac{t-\tau}{\tau}\Big). \tag{61}$$

Here we set $\tau = 0.002$. The model was then approximated in discrete-time with bin-size $\Delta = 0.001$. Spike trains were simulated in N = 2000 time-steps, and the filter was estimated by the two methods; about 250 spikes were used for inferring the filters in each simulation. Fig. 7 shows the estimated filters by the two methods. We see that these estimates agree quite well for this small value of σ.

5 Discussion

In this article, we investigated efficient computation of the MAP path in state-space models and applied these methods to parameter estimation and prediction of the sub-threshold voltage in the leaky IF neuron model. The key idea is that the Hessian matrix of the log-posterior density becomes a block tridiagonal matrix because of the Markov structure of the state-space model, which in turn allows us to exactly compute the MAP path in $O(Nd^3)$ time [7,12,13]. We note that the efficient computation is made possible by taking advantage of the banded nature of the Hessian, not explicitly the

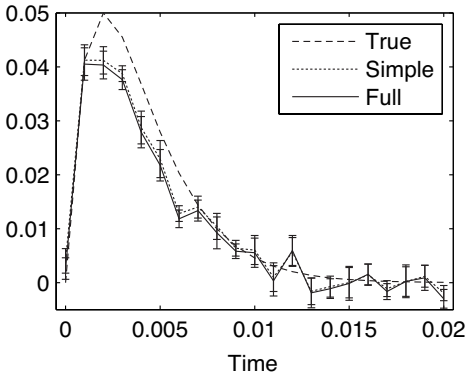

Fig. 7 Estimating an unknown stimulus filter. The true filter a_i (dashed line; eq. (60)) vs. the estimated filters provided by the full marginal likelihood (solid line) and by the simplified method (no log-determinant term; dotted line). The mean and standard error were calculated with 5 samples. Note that the two solutions are similar in this low-σ setting.

tridiagonal structure. Hence our methods can be applied more generally whenever the Hessian is banded [8].

An alternative method to compute the MAP path is to solve the Euler-Lagrange equation (e.g., eq. (14)) numerically by a shooting method or a relaxation method. In a shooting method, the problem is replaced by a search over the initial condition, to find a solution of the Euler-Lagrange equation that satisfies the boundary conditions. For the point process observation model, the spike observation is incorporated in the Euler-Lagrange equation as a discontinuous perturbation, as in (14), and the solution of the differential equation becomes unstable (in the sense that it is sensitive to the initial conditions). The shooting method is inefficient in such cases [11]. The idea of relaxation methods, on the other hand, is that the original differential equation is approximated to a finite difference equation, and then linearized about a guessed solution to obtain a system of linear equations whose solution provides a modification of the current solution. Since the Euler-Lagrange equation corresponds to the gradient of the log-posterior distribution equated to zero, the relaxation method and the tridiagonal Newton-Raphson method are in fact equivalent here. As we have seen here, the direct optimization approach (and corresponding probabilistic Laplace approximation) leads to a more general and flexible framework for this model.

The tridiagonal Newton-Raphson method has a helpful interpretation in terms of an iterative Kalman filter which is worth noting here [7,12,13]. Consider a linear Gaussian state-space model whose joint probability distributions of the state and observation model are, respectively, given by

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, C), \tag{62}$$

$$p(\boldsymbol{y}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{y}; \boldsymbol{x}, H), \tag{63}$$

where $\boldsymbol{x} = (x_1, x_2, \ldots, x_N)^T$ is the state vector and $\boldsymbol{y} = (y_1, y_2, \ldots, y_N)^T$ is the observation vector. It follows that the conditional mean of the state, $E(\boldsymbol{x}|\boldsymbol{y})$, is computed as

$$E(\boldsymbol{x}|\boldsymbol{y}) = (C^{-1} + H^{-1})^{-1}(H^{-1}\boldsymbol{y} + C^{-1}\boldsymbol{\mu}). \tag{64}$$

The Kalman filter and smoother compute the conditional mean in a recursive and computationally efficient way for a linear Gaussian state space model.

On the other hand, the Newton-Raphson updating step for computing the MAP path of a general state-space mode is given by

$$\hat{\boldsymbol{x}}^{(l+1)} = \hat{\boldsymbol{x}}^{(l)} - (A^{-1} + C^{-1})^{-1}[\nabla_x \log p(\boldsymbol{y}|\hat{\boldsymbol{x}}^{(l)}) + \nabla_x \log p(\hat{\boldsymbol{x}}^{(l)})], \tag{65}$$

where $A = -[\nabla\nabla_x \log p(\boldsymbol{y}|\hat{\boldsymbol{x}}^{(l)})]^{-1}$ and C is the covariance matrix of $p(\boldsymbol{x})$. Applying the second-order approximation both to $\log p(\boldsymbol{y}|\boldsymbol{x})$ about the current estimate $\hat{\boldsymbol{x}}^{(l)}$ and $\log p(\boldsymbol{x})$, which corresponds to a linear Gaussian approximation, the Newton-Raphson updating step becomes

$$\hat{\boldsymbol{x}}^{(l+1)} = (A^{-1} + C^{-1})^{-1}(A^{-1}\boldsymbol{z} + C^{-1}\boldsymbol{\mu}), \tag{66}$$

where $\boldsymbol{\mu}$ is the mean vector of $p(\boldsymbol{x})$ and $\boldsymbol{z} = \hat{\boldsymbol{x}}^{(l)} + A\nabla_x \log p(\boldsymbol{y}|\hat{\boldsymbol{x}}^{(l)})$. Thus each Newton iteration corresponds to the Kalman filter and smoother with $\boldsymbol{y} = \boldsymbol{z}$ and $H = A$ in (64), where the coefficients of the Kalman model are updated once per Newton iteration as we update the Hessian and gradient of $\log p(\boldsymbol{y}|\boldsymbol{x})$ at $\hat{\boldsymbol{x}}^{(l)}$. Indeed, in the case of very large Nd, we can use standard Kalman-smoother code to compute our Newton-Raphson search direction in a memory-efficient way (since we do not need to store A in memory explicitly). Of course, in the case of linear Gaussian observations and transitions, the Newton-Raphson algorithm terminates after one step, since the Hessian remains constant with each iteration in this case, and we are left with the standard Kalman filter.

We compared the tridiagonal Newton-Raphson method with two other methods: the point process filter/smoother and the expectation propagation method. The point process filter is a nonlinear- and non-Gaussian recursive Bayesian filter, in which the filtered distribution is approximated to a Gaussian at each recursion step. When the state-transition density is linear Gaussian, all conditional densities become Gaussian, and thus filtering and smoothing can be done by the standard Kalman procedure in $O(N)$. The main difference between the tridiagonal Newton-Raphson method and the point process filter/smoother is that the former computes the exact MAP path, while the latter approximates the MAP path (or posterior expectation), using iterative "local" approximations at each time point t. These approximations are valid in two limiting situations [42]: the "low-information" limit in the case of linear-Gaussian prior dynamics, where the signal-to-noise of the spiking response is poor and the non-Gaussian observation terms $p(y_i|x_i)$ vary weakly as a function of x_i, making the Gaussian prior $p(x_i)$ dominant; and the "high-information" limit, where the posterior $p(x_i|\boldsymbol{y})$ becomes very sharply peaked around the mode and a local Laplace approximation is valid.

In cases where this Laplace approximation is justified, the MAP path is a good approximation of the posterior mean, since the mode and the mean of a Gaussian are identical. Indeed, when the variance of the posterior is scaled by a parameter n^{-1} where n is a large parameter indexing the informativeness of the observed data, the MAP estimate provides the first-order approximation for the posterior mean with respect to n^{-1} [43]. However, in cases where the posterior distribution is far from Gaussian, the Laplace approximation can fail, resulting in a large average error in the MAP estimate. In such a case, the posterior mean can provide a better estimate, since this is the optimal Bayesian estimate under squared error loss. Indeed, in the numerical study on the leaky IF model here, we saw that the EP approximation becomes better than the MAP path especially when σ is large.

To summarize, we have decribed an efficient algorithm for computing the exact MAP path and for estimating the parameters in state-space models. While we have focused on the integrate-and-fire model here as a concrete example, these ideas can be exploited in a wide variety of neural settings. We discuss a number of further examples in [6].

A Point process filter and smoother

A simple version of the point process filter approximates the filtered distribution (26) to a Gaussian centered by its mode [2]. Let $x_{i|i}$ and $V_{i|i}$ be the (approximate) mode and covariance matrix for the filtered distribution (26), and $x_{i|i-1}$ and $V_{i|i-1}$ be the mode and covariance matrix for the predictive distribution (27) at time i. Let $l(x_i) = \log\{p(y_i|x_i)p(x_i|y_{1:i-1})\}$. The filtered distribution is then approximated to a Gaussian whose mean and covariance are $x_{i|i} = \arg\max_{x_i} l(x_i)$ and $V_{i|i} = -[\nabla\nabla_{x_i} l(x_{i|i})]^{-1}$, respectively. When the state-transition density is linear Gaussian, $p(x_i|x_{i-1}) = \mathcal{N}(F_i x_{i-1}, Q_i)$, the predictive distribution (27) is also Gaussian, whose mean and covariance are computed as

$$x_{i|i-1} = F_i x_{i-1|i-1}, \tag{67}$$

$$V_{i|i-1} = F_i V_{i-1|i-1} F_i^T + Q_i. \tag{68}$$

Since the filtered and predictive distributions are Gaussian, the smoothing distribution (28) is also Gaussian, which can be computed by the standard Kalman smoother [3]. Let $x_{i|N}$ and $V_{i|N}$ be the mean and covariance the smoothing distribution at time i. The recursive smoothing equation corresponding to (28) is given by

$$x_{i|N} = x_{i|i} + V_{i|i} F_i V_{i+1|i}^{-1}(x_{i+1|N} - x_{i+1|i}), \tag{69}$$

$$V_{i|N} = V_{i|i} + V_{i|i} F_i V_{i+1|i}^{-1}(V_{i+1|N} - V_{i+1|i}) V_{i|i}^{-1} F_i^T V_{i+1|i}. \tag{70}$$

There are now several versions of the point process filter depending on the choice of the mean and variance of the approximate filtered distribution. In [5], the filtered distribution at each time step i is approximated to a Gaussian by expanding its logarithm in a Taylor series about $x_{i|i-1}$ up to the second-order term, which results in a simpler algorithm. [44] proposed a more sophisticated method by utilizing the fully exponential Laplace approximation [43], which achieves second-order accuracy in approximating the posterior expectation.

For the leaky IF model with hard-threshold, the standard Taylor-series-based recursions [2] do not apply (due to the discontinuity of $\log p(y_i|x_i)$), and therefore we have not included comparisons to the point-process smoother in Figs. 4- 6. However, it is worth noting that in this case the filtered distribution (26) can be approximated recursively as a truncated Gaussian defined on $(-\infty, x_{th}]$, and hence the approximate mean and variance can be obtained analytically; we found that this moment-matching method behaves similarly to the EP method (this is unsurprising, since EP is also based on a moment-matching procedure; data not shown).

B Gaussian quadrature in EP algorithm

The expectation of a function of x_i, $f(x_i)$, with respect to $p(x_i|y_{1:N})$ in (38) is expressed as

$$\begin{aligned} E_i[f(x_i)] &= \int\int f(x_i)p(x_{i-1}, x_i|y_{1:N})dx_{i-1}dx_i \\ &\propto \int f(x_i)p(y_i|x_i)\beta_i(x_i)\left[\int \alpha_{i-1}(x_{i-1})p(x_i|x_{i-1})dx_{i-1}\right]dx_i \\ &\equiv \int f(x_i)p(y_i|x_i)\beta_i(x_i)g(x_i)dx_i, \end{aligned} \tag{71}$$

where

$$g(x_i) = \int \alpha_{i-1}(x_{i-1})p(x_i|x_{i-1})dx_{i-1} \tag{72}$$

is Gaussian since $\alpha_{i-1}(x_{i-1})$ and $p(x_i|x_{i-1})$ are also Gaussian. By introducing the Laplace approximation, $p_L(x_i) \equiv \mathcal{N}(m, v) \approx p(y_i|x_i)\beta_i(x_i)g(x_i)$, as a proposal distribution, the expectation can be expressed as

$$\begin{aligned} E_i[f(x_i)] &\propto \int p_L(x_i)\left[\frac{f(x_i)p(y_i|x_i)\beta_i(x_i)g(x_i)}{p_L(x_i)}\right]dx_i \\ &\equiv \int p_L(x_i)F(x_i)dx_i. \end{aligned} \tag{73}$$

After a linear change of variable, $x_i = \sqrt{v}u + m$, we have the standard form of the Gauss-Hermite quadrature,

$$\begin{aligned} E_i[f(x_i)] &\propto \int e^{-\frac{u^2}{2}} F(\sqrt{v}u + m)du \\ &\approx \sum_{l=1}^{n} w_l F(\sqrt{v}u_l + m), \end{aligned} \tag{74}$$

where the weights w_l and evaluation points u_l are chosen according to a quadrature rule. The advantages of this method is that it requires only an inner product once the weights and evaluation points are calculated. (These only have to be computed once.) The expectation of $f(x_{i-1})$ with respect to $p(x_{i-1}|y_{1:N})$ in (39) can be computed in the same way.

For the leaky IF model with hard-threshold, the observation model is given by the step-function, and thus the integral in (73) becomes

$$\int_{-\infty}^{\infty} p(y_i = 0|x_i)\ldots dx_i = \int_{-\infty}^{x_{th}} \ldots dx_i. \tag{75}$$

As a result, the expectation (73) is reduced to the integral over a truncated Gaussian, which can be computed analytically.

Acknowledgements We thank Y. Ahmadian, R. Kass, M. Nikitchenko, K. Rahnama Rad, M. Vidne and J. Vogelstein for helpful conversations and comments. SK is supported by NIH grants R01 MH064537, R01 EB005847 and R01 NS050256. LP is supported by NIH grant R01 EY018003, an NSF CAREER award, and a McKnight Scholar award.

References

1. Paninski, L., Brown, E.N., Iyengar, S., Kass, R.E.: Stochastic Methods in Neuroscience, chap. Statistical analysis of neuronal data via integrate-and-fire models. Oxford University Press, Oxford (2008)
2. Brown, E.N., Frank, L.M., Tang, D., Quirk, M.C., Wilson, M.A.: 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 (1998)
3. Smith, A.C., Brown, E.N.: Estimating a state-space model from point process observations. Neural Computation **15**, 965–991 (2003)
4. Brockwell, A.E., Rojas, A.L., Kass, R.E.: Recursive Bayesian decoding of motor cortical signals by particle filtering. Journal of Neurophysiology **91**, 1899–1907 (2004)
5. Eden, U.T., Frank, L.M., Barbieri, R., Solo, V., Brown, E.N.: Dynamic analyses of neural encoding by point process adaptive filtering. Neural Computation **16**, 971–998 (2004)
6. Paninski, L., Ahmadian, Y., Ferreira, D., Koyama, S., Rahnama, K., Vidne, M., Vogelstein, J., Wu, W.: A new look at state-space models for neural data. Submitted (2009)
7. Fahrmeir, L., Kaufmann, H.: On Kalman filtering, posterior mode estimation and Fisher scoring in dynamic exponential family regression. Metrika **38**, 37–60 (1991)
8. Pillow, J., Ahmadian, Y., Paninski, L.: Model-based decoding, information estimation, and change-point detection in multi-neuron spike trains. Under review, Neural Computation (2008)

9. Koyama, S., Shinomoto, S.: Empirical Bayes interpretations for random point events. Journal of Physics A: Mathematical and General **38**, L531–L537 (2005)
10. Paninski, L.: The most likely voltage path and large deviations approximations for integrate-and-fire neurons. Journal of Computational Neuroscience **21**, 71–87 (2006)
11. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes in C. Cambridge University Press, Cambridge (1992)
12. Davis, R.A., Rodriguez-Yam, G.: Estimation for state-space models based on a likelihood approximation. Statistica Sinica **15**, 381–406 (2005)
13. Jungbacker, B., Koopman, S.J.: Monte Carlo estimation for nonlinear non-Gaussian state-space models. Biometrika **94**, 827–839 (2007)
14. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
15. Minka, T.: Expectation propagation for approximate Bayesian inference. Uncertainty in Artificial intelligence **17** (2001)
16. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society: B **79**, 1–38 (1977)
17. Ahmed, N.U.: Linear and Nonlinear Filtering for Scientists and Engineers. World Scientific, Singapore (1998)
18. West, M., Harrison, J.P., Migon, H.S.: Dynamic generalized linear models and Bayesian forcasting. Journal of the American Statistical Association **80**, 73–83 (1985)
19. Snyder, D.L.: Random Point Processes. John Wiley & Sons, Inc., New York (1975)
20. Huys, Q., Ahrens, M., Paninski, L.: Efficient estimation of detailed single-neuron models. Journal of Neurophysiology **96**, 872–890 (2006)
21. Izhikevich, E.M.: Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting. The MIT press, Cambridge (2007)
22. Moehlis, J., Shea-Brown, E., Rabitz, H.: Optimal inputs for phase models of spiking neurons. ASME Journal of Computational and Nonlinear Dynamics **1**, 358–367 (2006)
23. Koyama, S., Shimokawa, T., Shinomoto, S.: Phase transitions in the estimation of event rate: a path integral analysis. Journal of Physics A: Mathematical and General **40**, F383–F390 (2007)
24. Badel, L., Richardson, M., Gerstner, W.: Dependence of the spike-triggered average voltage on membrane response properties. Neurocomputing **69**, 1062–1065 (2005)
25. Paninski, L.: The spike-triggered average of the integrate-and-fire cell driven by Gaussian white noise. Neural Computation **18**, 2592–2616 (2006)
26. Fahrmeir, L., Tutz, G.: Multivariate Statistical Modelling Based on Generalized Linear Models. Springer (1994)
27. Paninski, L.: Maximum likelihood estimation of cascade point-process neural encoding models. Network: Computation in Neural Systems **15**, 243–262 (2004)
28. Paninski, L., Pillow, J., Simoncelli, E.: Maximum likelihood estimation of a stochastic integrate-and-fire neural model. Neural Computation **16**, 2533–2561 (2004)
29. Rabiner, L.: A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE **77**, 257–286 (1989)
30. Roweis, S., Ghahramani, Z.: A unifying review of linear Gaussian models. Neural Computation **11**, 305–345 (1999)
31. Heskes, T., Zoeter, O.: Expectation propagation for approximate inference in dynamic Bayesian networks. In: A. Darwiche, N. Friedman (eds.) Uncertainty in Artificial Intelligence: Proceedings of the Eighteenth Conference (UAI-2002), pp. 216–233. Morgan Kaufmann Publishers, San Francisco, CA (2002)
32. Yu, B.M., Shenoy, K.V., Sahani, M.: Expectation propagation for inference in non-linear dynamical models with Poisson observations. In: Proceedings of the Nonlinear Statistical Signal Processing Workshop. IEEE (2006)
33. Ypma, A., Heskes, T.: Novel approximations for inference in nonlinear dynamical systems using expectation propagation. Neurocomputing **69**, 85–99 (2005). URL http://dx.doi.org/10.1016/j.neucom.2005.02.020
34. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. The MIT Press, Cambridge (2006)
35. Paninski, L.: Log-concavity results on Gaussian process methods for supervised and unsupervised learning. Advances in Neural Information Processing Systems **17** (2005)
36. Rybicki, G., Hummer, D.: An accelerated lambda iteration method for multilevel radiative transfer, appendix b: Fast solution for the diagonal elements of the inverse of a tridiagonal matrix. Astronomy and Astrophysics **245**, 171 (1991)

37. Rybicki, G.B., Press, W.H.: Class of fast methods for processing irregularly sampled or otherwise inhomogeneous one-dimensional data. Phys. Rev. Lett. **74**(7), 1060–1063 (1995). DOI 10.1103/PhysRevLett.74.1060
38. Asif, A., Moura, J.: Block matrices with l-block banded inverse: Inversion algorithms. IEEE Transactions on Signal Processing **53**, 630–642 (2005)
39. Abarbanel, H., Creveling, D., Jeanne, J.: Estimation of parameters in nonlinear systems using balanced synchronization. Phys. Rev. E **77**, 016,208 (2008)
40. Salakhutdinov, R., Roweis, S.T., Ghahramani, Z.: Optimization with EM and expectation-conjugate-gradient. International Conference on Machine Learning **20**, 672–679 (2003)
41. Olsson, R.K., Petersen, K.B., Lehn-Schioler, T.: State-space models: from the EM algorithm to a gradient approach. Neural Computation **19**, 1097–1111 (2007)
42. Ahmadian, Y., Pillow, J., Paninski, L.: Efficient Markov Chain Monte Carlo methods for decoding population spike trains. Under review, Neural Computation (2008)
43. Tierney, L., Kass, R.E., Kadane, J.B.: Fully exponential Laplace approximation to posterior expectations and variances. Journal of the American Statistical Association **84**, 710–716 (1989)
44. Koyama, S., Pérez-Bolde, L.C., Shalizi, C.R., Kass, R.E.: Approximate methods for state-space models (2008). Under review