Estimating a State-Space Model from Point Process Observations

Anne C. Smith
asmith@neurostat.mgh.harvard.edu
Neuroscience Statistics Research Laboratory, Department of Anesthesia and Critical Care, Massachusetts General Hospital, Boston, MA 02114, U.S.A.

Emery N. Brown
brown@neurostat.mgh.harvard.edu
Neuroscience Statistics Research Laboratory, Department of Anesthesia and Critical Care, Massachusetts General Hospital, Boston, MA 02114, U.S.A., and Division of Health Sciences and Technology, Harvard Medical School/Massachusetts Institute of Technology, Cambridge, MA 02139, U.S.A.

A widely used signal processing paradigm is the state-space model. The state-space model is defined by two equations: an observation equation that describes how the hidden state or latent process is observed and a state equation that defines the evolution of the process through time. Inspired by neurophysiology experiments in which neural spiking activity is induced by an implicit (latent) stimulus, we develop an algorithm to estimate a state-space model observed through point process measurements. We represent the latent process modulating the neural spiking activity as a gaussian autoregressive model driven by an external stimulus. Given the latent process, neural spiking activity is characterized as a general point process defined by its conditional intensity function. We develop an approximate expectation-maximization (EM) algorithm to estimate the unobservable state-space process, its parameters, and the parameters of the point process. The EM algorithm combines a point process recursive nonlinear filter algorithm, the fixed interval smoothing algorithm, and the state-space covariance algorithm to compute the complete data log likelihood efficiently. We use a Kolmogorov-Smirnov test based on the time-rescaling theorem to evaluate agreement between the model and point process data. We illustrate the model with two simulated data examples: an ensemble of Poisson neurons driven by a common stimulus and a single neuron whose conditional intensity function is approximated as a local Bernoulli process.
1 Introduction

A widely used signal processing paradigm in many fields of science and engineering is the state-space model. The state-space model is defined by two equations: an observation equation that defines what is being measured or observed and a state equation that defines the evolution of the process through time. State-space models, also termed latent process models or hidden Markov models, have been used extensively in the analysis of continuous-valued data. For a linear gaussian observation process and a linear gaussian state equation with known parameters, the state-space estimation problem is solved using the well-known Kalman filter. Many extensions of this algorithm to both nongaussian, nonlinear state equations and nongaussian, nonlinear observation processes have been studied (Ljung & Söderström, 1987; Kay, 1988; Kitagawa & Gersh, 1996; Roweis & Ghahramani, 1999; Ghahramani, 2001). An extension that has received less attention, and the one we study here, is the case in which the observation model is a point process.

This work is motivated by a data analysis problem that arises from a form of the stimulus-response experiments used in neurophysiology. In the stimulus-response experiment, a stimulus under the control of the experimenter is applied, and the response of the neural system, typically the neural spiking activity, is recorded. In many experiments, the stimulus is explicit, such as the position of a rat in its environment for hippocampal place cells (O’Keefe & Dostrovsky, 1971; Wilson & McNaughton, 1993), velocity of a moving object in the visual field of a fly H1 neuron (Bialek, Rieke, de Ruyter van Steveninck, & Warland, 1991), or light stimulation for retinal ganglion cells (Berry, Warland, & Meister, 1997). In other experiments, the stimulus is implicit, such as for a monkey executing a behavioral task in response to visual cues (Riehle, Grün, Diesmann, & Aertsen, 1997) or trace conditioning in the rabbit (McClelland, Weible, & Disterhoft, 2001). The neural spiking activity in implicit stimulus experiments is frequently analyzed by binning the spikes and plotting the peristimulus time histogram (PSTH). When several neurons are recorded in parallel, cross-correlations or unitary events analysis (Riehle et al., 1997; Grün, Diesmann, Grammont, Riehle, & Aertsen, 1999) have been used to analyze synchrony and changes in firing rates. Parametric model-based statistical analysis has been performed for the explicit stimuli of hippocampal place cells using position data (Brown, Frank, Tang, Quirk, & Wilson, 1998). However, specifying a model when the stimulus is latent or implicit is more challenging. State-space models suggest an approach to developing a model-based framework for analyzing stimulus-response experiments when the stimulus is implicit.

We develop an approach to estimating state-space models observed through a point process. We represent the latent (implicit) process modulating the neural spiking activity as a gaussian autoregressive model driven by an external stimulus. Given the latent process, neural spiking activity...
is characterized as a general point process defined by its conditional intensity function. We will be concerned here with estimating the unobservable state or latent process, its parameters, and the parameters of the point process model. Several approaches have been taken to the problem of simultaneous state estimation and model parameter estimation, the latter being termed system identification (Roweis & Ghahramani, 1999). In this article, we present an approximate expectation-maximization (EM) algorithm (Dempster, Laird, & Rubin, 1977) to solve this simultaneous estimation problem. The approximate EM algorithm combines a point process recursive nonlinear filter algorithm, the fixed interval smoothing algorithm, and the state-space covariance algorithm to compute the complete data log likelihood efficiently. We use a Kolmogorov-Smirnov test based on the time-rescaling theorem to evaluate agreement between the model and point process data. We illustrate the algorithm with two simulated data examples: an ensemble of Poisson neurons driven by a common stimulus and a single neuron whose conditional intensity function is approximated as a local Bernoulli process.

2 Theory

2.1 Notation and the Point Process Conditional Intensity Function.
Let \((0, T]\) be an observation interval during which we record the spiking activity of \(C\) neurons. Let \(0 < u_{i1} < u_{i2} < \ldots, < u_{iJ_i} \leq T\) be the set of \(J_i\) spike times (point process observations) from neuron \(c\) for \(c = 1, \ldots, C\). For \(t \in (0, T]\), let \(N_{0,t}^c\) be the sample path of the spike times from neuron \(c\) in \((0, t]\). It is defined as the event \(N_{0,t}^c = \{0 < u_{i1} < u_{i2}, \ldots, u_{ij} \leq t \cap N^c(t) = j\}\), where \(N^c(t)\) is the number of spikes in \((0, t]\) and \(j \leq J_i\). The sample path is a right continuous function that jumps 1 at the spike times and is constant otherwise (Snyder & Miller, 1991). This function tracks the location and number of spikes in \((0, t]\) and therefore contains all the information in the sequence of spike times. Let \(N_{0,t} = \{N_{0,t}^1, \ldots, N_{0,t}^C\}\) be the ensemble spiking activity in \((0, t]\).

The spiking activity of each neuron can depend on the history of the ensemble, as well as that of the stimulus. To represent this dependence, we define the set of stimuli applied in \((0, t]\) as \(S_{0,t} = \{0 < s_1 < \ldots, < s_{\ell} \leq t\}\). Let \(H_t = \{N_{0,t}, S_{0,t}\}\) be the history of all \(C\) neurons up to and including time \(t\). To define a probability model for the neural spiking activity, we define the conditional intensity function for \(t \in (0, T]\) as (Cox & Isham, 1980; Daley & Vere-Jones, 1988):

\[
\lambda_c(t \mid H_t) = \lim_{\Delta \to 0} \frac{\Pr(N_{0,t+\Delta}^c - N_{0,t}^c = 1 \mid H_t)}{\Delta}.
\] (2.1)

The conditional intensity function is a history-dependent rate function that generalizes the definition of the Poisson rate (Cox & Isham, 1980; Daley &
Vere-Jones, 1988). If the point process is an inhomogeneous Poisson process, the conditional intensity function is \( \lambda_c(t \mid H_t) = \lambda_c(t) \). It follows that \( \lambda_c(t \mid H_t) \Delta \) is the probability of a spike in \([t, t + \Delta)\) when there is history dependence in the spike train. In survival analysis, the conditional intensity is termed the hazard function because, in this case, \( \lambda_c(t \mid H_t) \Delta \) measures the probability of a failure or death in \([t, t + \Delta)\) given that the process has survived up to time \( t \) (Kalbfleisch & Prentice, 1980).

### 2.2 Latent Process Model, Sample Path Probability Density, and the Complete Data Likelihood

It is possible to define the latent process in continuous time. However, to simplify the notation for our filtering, smoothing, and EM algorithms, we assume that the latent process is defined on a discrete set of evenly spaced lattice points. To define the lattice, we choose \( K \) large, and divide \((0, T]\) into \( K \) intervals of equal width \( \Delta = T/K \), so that there is at most one spike per interval. The latent process model is evaluated at \( k\Delta \) for \( k = 1, \ldots, K \). We also assume the stimulus inputs can be measured at a resolution of \( \Delta \).

We define the latent model as the first-order autoregressive model,

\[
    x_k = \rho x_{k-1} + \alpha I_k + \epsilon_k, \tag{2.2}
\]

where \( x_k \) is the unknown state at time \( k\Delta \), \( \rho \) is a correlation coefficient, \( I_k \) is the indicator function that is 1 if there is a stimulus at \( k\Delta \) and zero otherwise, \( \alpha \) modulates the effect of the stimulus on the latent process, and \( \epsilon_k \) is a gaussian random variable with mean zero and variance \( \sigma^2_\epsilon \). While more complex latent process models can certainly be defined, equation 2.2 is adequate to illustrate the essential features of our algorithm.

The joint probability density of the latent process is

\[
p(x \mid \rho, \alpha, \sigma^2_\epsilon) = \left[ \frac{(1 - \rho^2)}{2\pi \sigma^2_\epsilon} \right]^{\frac{K}{2}} \times \exp \left\{ -\frac{1}{2} \left[ \frac{(1 - \rho^2)}{\sigma^2_\epsilon} \lambda_0^2 \right. \right. \\
\left. \left. + \sum_{k=1}^{K} \frac{(x_k - \rho x_{k-1} - \alpha I_k)^2}{\sigma^2_\epsilon} \right] \right\}. \tag{2.3}
\]

where \( x = (x_0, x_1, \ldots, x_K) \).

We assume that the conditional intensity function is \( \lambda_c(k\Delta \mid x_k, H_k, \theta^*_c) \), where \( \theta^*_c \) is an unknown parameter. We can express the joint probability density of the sample path of neuron \( c \) conditional on the latent process as (Barbieri, Quirk, Frank, Wilson, & Brown, 2001; Brown, Barbieri, Ventura,
State Estimation from Point Process Observations

Kass, & Frank, 2002)

\[
p(N_0^C | x, H^C_T, \theta^*_c) = \exp \left[ \int_0^T \log \lambda_c(u | x(u), H^C_u, \theta^*_c) \ dN_c(u) \right. \\
\left. - \int_0^T \lambda_c(u | x(u), H^C_u, \theta^*_c) \ du \right],
\]

(2.4)

where \( dN_c(u) = 1 \) if there is a spike at \( u \) from neuron \( c \) and 0 otherwise. Under the assumption that the neurons in the ensemble are conditionally independent given the latent process, the joint probability density of the sample paths of the ensemble is

\[
p(N_0^C, T | x, H_T, \theta^*) = C \prod_{c=1}^C p(N_c^C | x, H^C_T, \theta^*_c),
\]

(2.5)

where \( \theta^* = (\theta^*_1, \ldots, \theta^*_C) \).

2.3 Parameter Estimation: Expectation-Maximization Algorithm. To illustrate the algorithm, we choose a simple form of the conditional intensity function. That is, we take the conditional intensity function for neuron \( c \) as

\[
\lambda_c(k \Delta) = \exp(\mu_c + \beta_c x),
\]

(2.6)

where \( \mu_c \) is the log of the background firing rate and \( \beta_c \) is its gain parameter that governs how much the latent process modulates the firing rate of this neuron. Here we have \( \theta^*_c = (\mu_c, \beta_c) \). Equations 2.3 and 2.6 define a doubly stochastic point process (Cox & Isham, 1980). If we condition on the latent process, then equation 2.6 defines an inhomogeneous Poisson process. Under this model, all the history dependence is through the stimulus. We let \( \theta = (\rho, \alpha, \sigma^2_\varepsilon, \theta^*) \). Because our objective is to estimate the latent process, \( x \), and to compute the maximum likelihood estimate of the model parameter, \( \theta \), we develop an EM algorithm (Dempster et al., 1977). In our EM algorithm, we treat the latent process \( x \) as the missing or unobserved quantity. The EM algorithm requires us to maximize the expectation of the complete data log likelihood. It follows from equations 2.3 and 2.6 that the complete data likelihood for our model is

\[
p(N_0^C, x | \theta) = p(N_0^C, x | \theta^*) p(x | \rho, \alpha, \sigma^2_\varepsilon).
\]

(2.7)

2.3.1 E-Step. At iteration \( \ell + 1 \) of the algorithm, we compute in the E-step the expectation of the complete data log likelihood given \( H^C \), the ensemble
spiking activity and stimulus activity in \((0, T]\), and \(\theta^{(l)}\), the parameter estimate from iteration \(l\). By our notation convention in the previous section, since \(K\Delta = T\), \(H_K = H_T\) and

\[
Q(\theta | \theta^{(l)}) = E[\log[p(N_{0,T}, x | \theta)] \| H_K, \theta^{(l)}]
\]

\[
= E \left[ \sum_{k=0}^{K} \sum_{c=1}^{C} (dN_c(k\Delta)(\mu_c + \beta_c x_k + \log \Delta)
- \exp(\mu_c + \beta_c x_k)) \| H_K, \theta^{(l)} \right]
\]

\[
+ E \left[ \sum_{k=1}^{K} -\frac{1}{2} \frac{(x_k - \rho x_{k-1} - \alpha l_k)^2}{\sigma^2} \right]
\]

\[
- \frac{K}{2} \log 2\pi - \frac{K}{2} \log \sigma^2 \| H_K, \theta^{(l)} \right]
\]

\[
+ E \left[ \frac{1}{2} \log(1 - \rho^2) - \frac{1}{2} \frac{\chi^2_{0}(1 - \rho^2)}{\sigma^2} \| H_K, \theta^{(l)} \right].
\]

Upon expanding the right side of equation 2.8, we see that calculating the expected value of the complete data log likelihood requires computing the expected value of the latent process \(E[x_k \| H_K, \theta^{(l)}]\) and the covariances \(E[x_k^2 \| H_K, \theta^{(l)}]\) and \(E[x_kx_{k+1} \| H_K, \theta^{(l)}]\). We denote them as

\[
x_{k|K} = E[x_k \| H_K, \theta^{(l)}]
\]

\[
W_k = E[x_k^2 \| H_K, \theta^{(l)}]
\]

\[
W_{k,k+1} = E[x_kx_{k+1} \| H_K, \theta^{(l)}].
\]

for \(k = 1, \ldots, K\) where the notation \(k \| j\) denotes the expectation of the latent process at \(k\Delta\) given the ensemble spiking activity and the stimulus up to time \(j\Delta\). To compute these quantities efficiently, we decompose the E-step into three parts: a forward nonlinear recursive filter to compute \(x_{k|k}\); a backward, fixed interval smoothing (FIS) algorithm to estimate \(x_{k|K}\); and a state-space covariance algorithm to estimate \(W_k\) and \(W_{k,k+1}\). This approach for evaluating the complete data log likelihood was suggested first by Shumway and Stoffer (1982). They used the FIS but a more complicated form of the state-covariance algorithm. An alternative covariance algorithm was given in Brown (1987). The logic of this approach is to compute the forward mean and covariance estimates and combine them with the backward mean and covariance estimates to obtain equations 2.10 and 2.11. This approach is exact for linear gaussian latent process models and linear gaussian
observation processes. For our model, it will be approximate because our observations form a point process.

**E-Step I: Nonlinear Recursive Filter.** The following equations comprise a recursive nonlinear filtering algorithm to estimate $x_{k|k}$ and $\sigma_{k|k}^2$ using equation 2.6 as the conditional intensity. The algorithm is based on the maximum a posteriori derivation of the Kalman filter algorithm (Mendel, 1995; Brown et al., 1998). It recursively computes a gaussian approximation to the posterior probability density $p(x_k | H_k, \theta (\ell))$. The approximation is based on recursively computing the posterior mode $x_{k|k}$ and computing its variance $\sigma_{k|k}^2$ as the negative inverse of the second derivative of the log posterior probability density (Tanner, 1996). The nonlinear recursive algorithm is:

**(Observation Equation)**

$$p(dN(k\Delta) \mid x_k) = \prod_{c=1}^{C} \left[ \exp(\mu_c + \beta_c x_k) \Delta \right]^{\text{d}N^c(k\Delta)} \times \exp(-\exp(\mu_c + \beta_c x_k) \Delta)$$ (2.12)

**(One-Step Prediction)**

$$x_{k|k-1} = \rho x_{k-1|k-1} + \alpha_l_k$$ (2.13)

**(One-Step Prediction Variance)**

$$\sigma_{k|k-1}^2 = \rho^2 \sigma_{k-1|k-1}^2 + \sigma^2_c$$ (2.14)

**(Posterior Mode)**

$$x_{k|k} = x_{k|k-1} + \sigma_{k|k-1}^2 \sum_{c=1}^{C} \beta_c \left[ d\text{N}^c(k\Delta) - \exp(\mu_c + \beta_c x_{k|k}) \Delta \right]$$ (2.15)

**(Posterior Variance)**

$$\sigma_{k|k}^2 = - \left[ -(\sigma_{k|k-1}^2)^{-1} - \sum_{i=1}^{C} \beta_i^2 \exp(\mu_c + \beta_c x_{k|k}) \Delta \right]^{-1}$$ (2.16)

for $k = 1, \ldots, K$. The initial condition is $x_0$ and $\sigma_{0|0}^2 = \sigma_c^2(1 - \rho^2)^{-1}$. The algorithm is nonlinear because $x_{k|k}$ appears on the left and right of equation 2.15. The derivation of this algorithm for an arbitrary point process model is given in the appendix.

**E-Step II: Fixed Interval Smoothing (FIS) Algorithm.** Given the sequence of posterior mode estimates $x_{k|k}$ (see equation 2.15) and the variance $\sigma_{k|k}^2$ (see
equation 2.16) we use the fixed interval smoothing algorithm to compute $x_{k|K}$ and $\sigma^2_{k|k}$. The algorithm is (Mendel, 1995; Brown et al., 1998)

$$x_{k|K} = x_{k|k} + A_k(x_{k+1|K} - x_{k+1|k}), \quad (2.17)$$

where

$$A_k = \rho \sigma^2_{k|k}(\sigma^2_{k+1|k})^{-1} \quad (2.18)$$

and

$$\sigma^2_{k|K} = \sigma^2_{k|k} + A_k^2(\sigma^2_{k+1|K} - \sigma^2_{k+1|k}) \quad (2.19)$$

for $k = K - 1, \ldots, 1$ and initial conditions $x_{K|K}$ and $\sigma^2_{K|K}$.

**E-Step III: State-Space Covariance Algorithm.** The covariance estimate, $\sigma_{k,u|K}$, can be computed from the state-space covariance algorithm (de Jong & MacKinnon, 1988) and is given as

$$\sigma_{k,u|K} = A_k \sigma_{k+1,u|K} \quad (2.20)$$

for $1 \leq k \leq u \leq K$. It follows that the covariance terms required for the E-step are

$$W_{k,k+1} = \sigma_{k,k+1|K} + x_{k|k}x_{k+1|k} \quad (2.21)$$

and

$$W_k = \sigma^2_{k|k} + x^2_{k|k} \quad (2.22)$$

**2.3.2 M-Step.** In the M-step, we maximize the expected value of the complete data log likelihood in equation 2.8 with respect to $\theta^{(\ell+1)}$. In so doing, we obtain the following closed-form solutions for $\rho^{(\ell+1)}$, $\alpha^{(\ell+1)}$ and $\sigma^2_{e}^{(\ell+1)}$,

$$\left[ \begin{array}{c} \rho^{(\ell+1)} \\ \alpha^{(\ell+1)} \end{array} \right] = \left[ \sum_{k=1}^{K} W_{k-1} \sum_{k=1}^{K} x_{k-1|k} I_k \sum_{k=1}^{K} W_{k,k-1} \sum_{k=1}^{K} x_{k|k} I_k \right]^{-1} \left[ \sum_{k=1}^{K} W_{k,k-1} \sum_{k=1}^{K} x_{k|k} I_k \right] \quad (2.23)$$

$$\sigma^2_{e}^{(\ell+1)} = K^{-1} \left[ \sum_{k=1}^{K} W_k + \rho^{2(\ell+1)} \sum_{k=1}^{K} W_{k-1} + \alpha^{2(\ell+1)} \sum_{k=1}^{K} I_k \right]$$
\[-2 \rho^{(\ell+1)} \sum_{k=1}^{K} W_{k,k-1} - 2 \alpha^{(\ell+1)} \sum_{k=1}^{K} x_{k|k} I_{k} + 2 \rho^{(\ell+1)} \alpha^{(\ell+1)} \]

\[\times \sum_{k=1}^{K} x_{k-1|k} I_{k} + W_{0} \left( 1 - \rho^{2(\ell+1)} \right) \right], \tag{2.24}\]

where initial conditions for the latent process are estimated from \(x_{0}^{(\ell+1)} = \rho^{(\ell+1)} x_{1|1} \) and \(\sigma_{00}^{2(\ell+1)} = \sigma_{x}^{2(\ell+1)} (1 - \rho^{2(\ell+1)})^{-1} \). The closed-form solution for \(\rho^{(\ell+1)} \) in equation 2.23 is obtained by neglecting the last two terms in the expectation of the complete data log likelihood (see equation 2.8). This approximation means that we estimate \(\rho^{(\ell+1)} \) from the probability density of \(x_{1}, \ldots, x_{K} \) given \(x_{0} \) and the point process measurements instead of the probability density of \(x_{0}, \ldots, x_{K} \) given the point process measurements. Inclusion of the last two terms results in a cubic equation for computing \(\rho^{(\ell+1)} \), which is avoided by using the closed-form approximation. We report only the results of the closed-form solution in section 3 because we found that for our algorithms, the absolute value of the fractional difference between the two solutions was less than \(10^{-6} \) (i.e., \(|\text{cubic solution} - \text{closed form solution}|/\text{cubic solution} < 10^{-6} \).

The parameter \(\mu^{(\ell+1)}_{c} \) is estimated as

\[\mu^{(\ell+1)}_{c} = \log N^{c}(T) - \log \left( \sum_{k=1}^{K} \exp \left( \beta^{(\ell+1)}_{c} x_{k|k} + \frac{1}{2} \beta^{2(\ell+1)}_{c} \sigma^{2}_{k|k} \right) \Delta \right) \]. \tag{2.25}\]

whereas \(\beta^{(\ell+1)}_{c} \) is the solution to the nonlinear equation,

\[\sum_{k=1}^{K} dN^{c}(k \Delta) x_{k|k} = \exp \mu^{(\ell+1)}_{c} \left\{ \sum_{k=1}^{K} \exp \left( \beta^{(\ell+1)}_{c} x_{k|k} + \frac{1}{2} \beta^{2(\ell+1)}_{c} \sigma^{2}_{k|k} \right) \right\}

\[\times \left( x_{k|k} + \beta^{(\ell+1)}_{c} \sigma^{2}_{k|k} \right) \Delta \}. \tag{2.26}\]

which is solved by Newton’s method after substituting \(\mu^{(\ell+1)}_{c} \) from equation 2.26. The expectations needed to derive equations 2.25 and 2.26 were computed using the lognormal probability density and the approximation of \(p(x_{k} \mid H_{k}, \theta^{(\ell)}) \) as a gaussian probability density with mean \(x_{k|k} \) and variance \(\sigma^{2}_{k|k} \).
2.4 Assessing Model Goodness-of-Fit by the Time-Rescaling Theorem. The latent process and point process models, along with the EM algorithm, provide a model and an estimation procedure for computing the latent process and the parameter vector $\theta$. It is important to evaluate model goodness-of-fit, that is, determine how well the model describes the neural spike train data series data. Because the spike train models are defined in terms of an explicit point process model, we can use the time-rescaling theorem to evaluate model goodness-of-fit. To do this, we compute for each neuron the time-rescaled or transformed interspike intervals

$$\tau_j = \int_{u_{j-1}}^{u_j} \lambda(u | \hat{\theta}) \, du,$$

where the $u_j$s are the spike times from the neuron and $\lambda(t | \theta)$ is the conditional intensity function in equation 2.6 evaluated at the maximum likelihood estimate $\hat{\theta}$ for $j = 1, \ldots, J$, where we have dropped the subscript $c$ to simplify notation. The $u_j$s are a point process with a well-defined conditional intensity function and, hence, by the time-rescaling theorem, the $\tau_j$s are independent, exponential random variables with a unit rate (Barbieri et al., 2001; Brown et al., 2002). Under the further transformation $z_j = 1 - \exp(-\tau_j)$, the $z_j$s are independent, uniform random variables on the interval $(0,1)$. Therefore, we can construct a Kolmogorov-Smirnov (K-S) test to measure agreement between the $z_j$s and the uniform probability density (Barbieri et al., 2001; Brown et al., 2002). First, we order the $z_j$s from the smallest to the largest value. Then we plot values of the cumulative distribution function of the uniform density defined as $b_j = j - 1/2$ for $j = 1, \ldots, J$ against the ordered $z_j$s. The points should lie on the 45 degree line. Because the transformation from the $u_j$s to the $z_j$s is one-to-one, a close agreement between the probability density of the $z_j$s and the uniform probability density on $(0,1)$ indicates close agreement between the (latent process-point process) model and the point process measurements. Hence, the time-rescaling theorem provides a direct means of measuring agreement between a point process or neural spike train time series and a probability model intended to describe its stochastic structure.

3 Applications

3.1 Example 1. Multiple Neurons Driven by a Common Latent Process. To illustrate our analysis paradigm, we simulate $C = 20$ simultaneously recorded neurons from the model described by equations 2.2 and 2.6. The time interval for the simulation was $T = 10$ seconds, and the latent process model parameters were $\rho = 0.99$, $\alpha = 3$, and $\sigma^2 = 10^{-3}$ with the implicit stimulus $I_k$ applied at 1 second intervals. The parameters for the conditional intensity function defining the observation process were the log of the background firing rate $\mu = -4.9$ for all neurons, whereas the gain coefficients
Table 1: True Parameter Values and EM Algorithm Parameter Estimates.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>True</th>
<th>Estimate</th>
<th>Parameter</th>
<th>True</th>
<th>Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\rho)</td>
<td>0.990</td>
<td>0.993</td>
<td>(\beta_{10})</td>
<td>1.079</td>
<td>1.130</td>
</tr>
<tr>
<td>(\alpha)</td>
<td>3.000</td>
<td>2.625</td>
<td>(\beta_{11})</td>
<td>1.008</td>
<td>1.039</td>
</tr>
<tr>
<td>(\mu)</td>
<td>-4.900</td>
<td>-5.105</td>
<td>(\beta_{12})</td>
<td>1.078</td>
<td>1.247</td>
</tr>
<tr>
<td>(\beta_1)</td>
<td>1.075</td>
<td>1.261</td>
<td>(\beta_{13})</td>
<td>1.009</td>
<td>1.031</td>
</tr>
<tr>
<td>(\beta_2)</td>
<td>0.989</td>
<td>1.060</td>
<td>(\beta_{14})</td>
<td>1.078</td>
<td>1.129</td>
</tr>
<tr>
<td>(\beta_3)</td>
<td>1.035</td>
<td>1.039</td>
<td>(\beta_{15})</td>
<td>0.980</td>
<td>1.121</td>
</tr>
<tr>
<td>(\beta_4)</td>
<td>0.973</td>
<td>1.101</td>
<td>(\beta_{16})</td>
<td>0.976</td>
<td>1.164</td>
</tr>
<tr>
<td>(\beta_5)</td>
<td>1.003</td>
<td>1.190</td>
<td>(\beta_{17})</td>
<td>0.990</td>
<td>1.234</td>
</tr>
<tr>
<td>(\beta_6)</td>
<td>0.992</td>
<td>1.132</td>
<td>(\beta_{18})</td>
<td>0.968</td>
<td>1.154</td>
</tr>
<tr>
<td>(\beta_7)</td>
<td>0.977</td>
<td>1.118</td>
<td>(\beta_{19})</td>
<td>0.956</td>
<td>0.979</td>
</tr>
<tr>
<td>(\beta_8)</td>
<td>1.027</td>
<td>1.176</td>
<td>(\beta_{20})</td>
<td>1.098</td>
<td>1.350</td>
</tr>
<tr>
<td>(\beta_9)</td>
<td>1.066</td>
<td>1.089</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\(\beta_c\) for the 20 neurons were chosen randomly on the interval \([0.9, 1.1]\). These parameter values for the latent and observation processes were chosen to give an approximate average firing rate of 10 Hz for each neuron. The neural spike trains were simulated using the time-rescaling theorem algorithm described in Brown et al. (2002), and the state equations in the EM algorithm were updated at \(\Delta = 1\) msec.

Using the EM algorithm described in the previous section, we fit the model in equations 2.2 and 2.6 simultaneously to the 20 simulated neural spike trains, assuming a fixed noise variance, \(\sigma^2 = 10^{-3}\). The convergence criteria for the algorithm were absolute changes of less than \(10^{-2}\) in consecutive iterates of the parameters and relative changes in the parameter iterates of less than \(10^{-3}\), that is, \(\text{old} - \text{new}/\text{old} < 10^{-3}\). The parameter estimates from the EM algorithm were in good agreement with the true values used to simulate the spike train data (see Table 1). In this case, the overestimates of the gain coefficients, \(\beta_c\), are offset by the underestimates of \(\alpha\) and \(\mu\).

Approximating the probability density of the state at \(k\Delta\) as the gaussian density with mean \(\hat{x}_k\) and variance, \(\hat{\sigma}^2_k\), it follows from equation 2.6 and the standard change of variables formula from probability theory that the probability density of the rate for neuron \(c\) at time \(k\Delta\) is the lognormal probability density defined as

\[
p(\lambda^c_k | \hat{\mu}_c, \hat{\beta}_c, \hat{x}_k) = (2\pi \hat{\sigma}^2_k)^{-\frac{1}{2}} \hat{\beta}_c (\lambda^c_k)^{-1} \times \exp \left( -\frac{1}{2} \left( \frac{\hat{\beta}_c^{-1}(\log \lambda^c_k - \hat{\mu}_c) - \hat{x}_k}{\hat{\sigma}_k} \right)^2 \right),
\]

where \(\hat{\mu}_c\) and \(\hat{\beta}_c\) are the EM algorithm estimates of \(\mu_c\) and \(\beta_c\) for \(c = 1, \ldots, 20\). The \(1 - \xi\) confidence limits for the rate are computed simply by using the
relation between the lognormal and standard gaussian probabilities to find the $\xi/2$ and $1 - \xi/2$ quantiles of the probability density in equation 3.1 for $\xi \in (0, 1)$. In our analyses, we take $\xi = 0.05$ and construct 95% confidence intervals.

To compare our model-based analysis with current practices for analyzing neural spike train data using empirical smoothing methods, we also estimated the rate function for each of the 20 neurons by dividing the number of spikes in a 100 msec window by 100 msec. The window was then shifted 1 msec to give the same temporal resolution as in our updating algorithms. Because the latent process drives all the neurons, we also estimated the population rate by averaging the rates across all 20 neurons. This is a commonly used empirical temporal smoothing algorithm for computing spike rate that does not make use of stimulus information in the estimation (Riehle et al., 1997; Grün et al., 1999; Wood, Dudchenko, & Eichenbaum, 1999).

The confidence limits of the model-based rate function give a good estimate of the true firing rate used to generate the spikes (see Figure 1). In particular, the estimates reproduce the magnitude and duration of the effect of the implicit stimulus on the spike firing rate. The population firing rate estimated using temporal smoothing across all neurons is misleading (see Figure 1, dot-dashed line) in that around the time of the stimuli, it has diminished amplitude and is spread out in time. Furthermore, if we smooth a single spike train without averaging across neurons, spurious peaks in the firing rate can be produced due to noise (see Figure 1, solid gray line). By using information about the timing of the stimulus, the model firing rate estimate follows the true rate function more closely.

The 95% confidence bounds for the state process estimated from the EM algorithm cover almost completely the time course of the true state process (see Figure 2). The true state lies sometimes outside the confidence limits in regions where there are very few spikes and, hence, little information about the latent process.

To assess how well the model fits the data, we apply the K-S goodness-of-fit tests based on the time-rescaling theorem as described in section 2.4 (see Figure 3). In Figures 3A, 3B, and 3C, the solid black line represents exact agreement between the model and spike data, and dotted lines represent 95% confidence limits. For 18 of 20 neurons, the model lies within the confidence limits, indicating a good fit to the data. In contrast, for 14 of the 20 neurons, the empirical estimate lies outside the confidence limits. For 6 of 20 neurons, both the model and empirical estimates lie within the confidence limits (see Figure 3B). In two cases, the model lies outside the confidence limits (see Figure 3C). In both cases where the model does not fit the data, the empirical estimate does not either. Thus, the model appears to give a considerably more accurate description of the spike train than the empirical estimate.
State Estimation from Point Process Observations

Figure 1: A 2000 millisecond segment of data from neuron 8 in the 20-neuron simulation study of latent process estimation by the EM algorithm from example 1. During this period, 26 spikes (vertical bars above abscissa) are concentrated around two stimulus events at 5000 msec and 6000 msec. The 95% confidence bands for the firing rate estimated by the latent model (dashed lines) cover the true rate (solid black line) used to generate the spikes most of the time. In contrast, when the firing rate for neuron 8 is estimated empirically (solid gray line) using a 100 msec smoothing window (i.e., by dividing the number of spikes in a 100 msec window by 100 msec), the stimuli appear diminished in amplitude and spread out in time. Furthermore, a spurious third peak in firing rate is produced approximately 200 ms after the true first peak. This third peak is avoided by using a smoothed estimate that is averaged over all cells (dash-dotted black line), but the EM algorithm estimate of the firing rate based on the model more accurately reproduces the features of the stimuli.

3.2 Example 2. Single Neuron Latent Stimulus-Response Model. A common practice is to use the binomial probability mass function as a local model for analyzing neural spike trains. We demonstrate how the parameters of a local Bernoulli model may be estimated using our EM algorithm. We generate a single spike train subject to multiple repeated stimuli over a period of approximately 1 minute. In this example, the conditional intensity is given by

\[
\lambda(k, \Delta) = \exp(\mu + x_k \Delta) \approx \frac{\exp(\mu + x_k \Delta)}{1 + \exp(\mu + x_k \Delta)} = p_k,
\]

(3.2)
Figure 2: True state (solid black line) and estimated 95% confidence limits (dashed lines) for the true state computed from the spike train observed in Figure 1 using the EM algorithm. The estimated confidence limits computed as $x_k | k \pm 1.96 \sigma_k | k$ fail to cover the true state when few spikes are discharged (e.g., near 5700 ms).

Figure 3: Facing page. Kolmogorov-Smirnov goodness-of-fit analyses based on the time-rescaling theorem for three representative neurons. Each panel is the K-S plots comparing the model rate estimate (dashed line), the empirical rate estimate (solid gray line), and the true rate (solid black line). In these figures, the 45 degree line in black represents an exact agreement between the model and the spike data. Dotted lines in each panel are the 95% confidence limits (see Brown et al., 2002, for details). Since the true rate was used to generate spikes, the true rate KS plot always lies within the confidence limits. (A) An example of the KS plot from 1 of the 18 out of the 20 neurons for which the model-based estimate of the KS plot was entirely within the confidence limits, indicating close agreement between the overall model fit and the simulated data. (B) An example of 1 of the 6 out of the 20 neurons for which the K-S plot based on the empirical rate estimate completely lies within the 95% confidence limits. (C) An example of 1 of the 2 out of the 20 neurons for which the K-S plot based on the empirical rate model did not remain in the 95% confidence bounds.
and spikes are generated with the local Bernoulli model,

\[ p(dN(k\Delta) \mid H_k) = p_k^{dN(k\Delta)} (1 - p_k)^{1 - dN(k\Delta)}, \]  

(3.3)

with parameters \( \mu = -4.6, \alpha = 4, \) and \( \rho = 0.8. \) The implicit stimulus is applied at 40 time points. We assume the noise \( \sigma^2 \) in the latent process is 0.2. In this example, we set \( \Delta = 5 \) msec. This generates a spike train with
Table 2: True Parameter Values and the EM Algorithm Parameter Estimates.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>$\rho$</th>
<th>$\sigma^2$</th>
<th>$\epsilon$</th>
<th>$\mu$</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>0.800</td>
<td>4.000</td>
<td>0.200</td>
<td>-4.600</td>
</tr>
<tr>
<td>Estimate</td>
<td>0.804</td>
<td>3.573</td>
<td>0.125</td>
<td>-4.404</td>
</tr>
</tbody>
</table>

an average firing rate over the minute period of approximately 10 Hz. The parameters, including the noise variance $\sigma^2$, are again estimated using the EM algorithm and are in good agreement with their true values (see Table 2).

As in the previous example, confidence intervals were computed for the true rate function by deriving the probability density function for $\lambda_k$ using the standard change of variable formula. Firing rates estimated by the model are compared with the empirical temporally smoothed rate computed by counting spikes in a 100 msec window and then dividing by 100 msec.

The firing rate (95% confidence limits) computed by the model compares favorably with the original rate used to generate the spikes (see Figure 4). The 95% confidence limits for the model estimate the magnitude and shape of the stimulus effect on the rate even when the stimulus has little obvious effect on the spike train. This is because the estimates for each stimulus’s effect are made based on the whole time series. In contrast, the temporally smoothed estimate of rate does not clearly identify stimulus effects on the spike train (see Figure 4, solid gray line). The choice of a 100 msec window for these simulated data appears too large as increases in firing rate at the stimuli are smoothed out. We can also make an overall estimate of the accuracy of the firing rates by comparing the model and the empirical firing rates with the true firing at the stimulus times, namely, at 40 points in time where we applied the stimulus during the 1 minute epoch of simulated data. The mean differences (standard error) between the empirical and model rates compared to the true rate at these times are $-113.9 (4.5)$ Hz and $-8.5 (4.4)$ Hz, respectively. Thus, the model provides a significantly better approximation to the true rate than the empirical method (Student’s $t$-test, $p < 10^{-6}$). The estimate of the state variable is found to compare well with the true state used to generate the spike train (see Figure 5).

Again we assess model goodness-of-fit using the K-S plot (see Figure 6). The fit of model lies within the 95% confidence limits except at very small quantiles and follows closely the curve for the true firing rate. In contrast, the fit for the empirical 100 msec temporal smoothing method lies outside the confidence limits for a large portion of the total time, indicating poor fit to the distribution over low and high percentiles.

3.3 An Extension to More General Point Process and State-Space Models. In the two simulated data examples we considered, the log conditional intensity function $\log \lambda_c(t | H_t)$ was a linear function of both the state vari-
Figure 4: Simulated spike train (vertical bars above the abcissa), true firing rate (solid black line) from the local Bernoulli model (see equation 3.2) in example 2, 95% confidence limits for the rate from the model-based EM algorithm (dashed lines), and empirical firing rate estimate (solid gray line) computed by temporal smoothing over the 100 msec window. In this time segment, two external stimuli are applied at 150 msec and 750 msec. The 95% confidence limits cover nearly everywhere the true rate function. Although the spike count does not obviously increase at these times, the algorithm estimates effectively the amplitude and duration of the stimulus because it uses information from the entire spike train. The 100 msec window for the empirical rate function appears too large as increases in firing rate at the stimuli are smoothed out.

\[ Q(\theta | \theta^{(i)}) = E \left[ \log p(N_{0,T}, x | \theta) \mid H_k, \theta^{(i)} \right] \]

\[ \approx E \left[ \sum_{k=0}^{K} \sum_{c=1}^{C} dN^c(k\Delta) \log \lambda_c(x_k | H_k, \theta) - \lambda_c(x_k | H_k, \theta) \Delta \mid H_k, \theta^{(i)} \right] \]

\[ + E[\log p(x | \theta) \mid H_k, \theta^{(i)}]. \]  

(3.4)
Figure 5: The true state (black line) and the model-based estimates of the 95% confidence intervals (dashed lines) computed using the EM algorithm for the local Bernoulli probability model corresponding to the spike train and rate function in Figure 4. As in example 1, Figure 5 shows that the 95% confidence limits cover the true state completely except when the neural spiking activity is low (around 700 msec).

where the last term in equation 3.4 is the sum of the last two terms on the right-hand side of equation 2.8. We assume \( \lambda_c(x_k | H_k, \theta) \) and \( \log \lambda_c(x_k | H_k, \theta) \) are twice differentiable functions that we denote generically as \( g(x_k) \).

To evaluate the first term on the right side of equation 3.4, it suffices to compute \( E[g(x_k | H_k, \theta) \| H_K, \theta^{(t)}] \). This can be accomplished by expanding \( g(x_k) \) in a Taylor series about \( \hat{x}_{k|K} \) and taking the expected value to obtain the approximation

\[
E[g(x_k | H_k, \theta) \| H_K, \theta^{(t)}] \approx g(\hat{x}_{k|K}) + \frac{1}{2} \sigma_{g_{k|K}}^2 g''(\hat{x}_{k|K}),
\]  

(3.5)

where \( g''(\hat{x}_{k|K}) \) is the second derivative of \( g(x_k) \) evaluated at \( \hat{x}_{k|K} \). The right-hand side of equation 3.5 is substituted into equation 3.4 to evaluate the E-step. The evaluation of the second term on the right of equation 3.4 proceeds as in the evaluation of the second and third terms on the right of equation 2.8.

If the log conditional intensity function is no longer a linear or approximately linear function of the parameters, the M-step takes the more
Fig. 6: Kolmogorov-Smirnov goodness-of-fit analysis for local Bernoulli probability model. The K-S plot for the EM algorithm model-based estimate (dashed line) lies almost entirely within the 95% confidence bounds (dotted lines) and follows closely the KS plot computed using the true firing rate (solid black line), which lies almost entirely on the 45 degree line of exact agreement. In contrast, the empirical rate estimate (solid gray line) lies mostly outside the confidence limits, suggesting that this model does not agree closely with the data.

The general form,
\[ \nabla Q(\theta | \theta^{(t)}) = 0, \] (3.6)
where the gradient in \( \nabla Q(\theta | \theta^{(t)}) \) is with respect to \( \theta \). The zero of \( \nabla Q(\theta | \theta^{(t)}) \) and, hence \( \theta^{(t+1)} \), has to be found using Newton’s method or another appropriate numerical optimization procedure. Similarly, the recursive filtering and smoothing algorithms generalize in a straightforward way when the dimension of the state-space model is greater than one (Brown et al., 1998). While these modifications increase the computational requirements of our algorithm, they illustrate how it can be applied to a wider range of point process and state-space models.

4 Discussion

We have presented a method for estimating from point process (spike train) data a conditional intensity (rate) function modulated by an unobservable
or latent continuous-valued state variable. The latent variable relates the effect of the external stimuli applied at specified times by the experimenter to the spike train rate function. We compute maximum likelihood estimates of the model parameters by the EM algorithm in which the E-step combines forward and backward point process filtering algorithms. The model performs better than smoothed histogram estimates of rate because it makes explicit use of the timing of the stimulus to analyze changes in background firing. Also, the model gives a more accurate description of the neural spike train as evaluated by the goodness-of-fit K-S test.

Several authors have discussed the analyses of state-space models in which the observation process is a point process. Diggle, Liang, and Zeger (1995) briefly mention state estimation from point process observations but no specific algorithms are given. MacDonald and Zucchini (1997) discuss state estimation for point processes without using the smoothing and filtering approach suggested here. West and Harrison (1997) define the state-space model implicitly and use the discount concept to construct an approximate forward filter. This approach is difficult to generalize (Fahrmeir, 1992). Kitagawa and Gersch (1996) described numerical algorithms to carry out state-space updating with forward recursion algorithms for binomial and Poisson observation processes. For the Poisson model in equation 2.6, Chan and Ledolter (1995) provided a computationally intensive Markov chain Monte Carlo EM algorithm to conduct the state updating and parameter estimation. The forward updating algorithms of Fahrmeir (1992), Fahrmeir and Tutz (1994), and Durbin and Koopman (2000) resemble most closely the ones we present, particularly, in the special case where the observation process is a point process from an exponential family, and the natural parameter is modeled as a linear function of the latent process. Both the examples we present follow these two special cases. The Fahrmeir forward recursion algorithm for example 1 with a single neuron is

\[ x_{k|k} = x_{k|k-1} + \frac{\sigma^2_{k|k-1}}{\lambda(k\Delta) + \sigma^2_{k|k-1}}[dN(k\Delta) - \lambda(k\Delta)] \]  

\[ \sigma^2_{k|k} = [\sigma^2_{k|k-1}]^{-1} + \lambda(k\Delta) \]  

(4.1)

(4.2)

whereas the Durbin and Koopman (2000) update is

\[ x_{k|k} = x_{k|k-1} + \frac{\sigma^2_{k|k-1}}{\lambda(k\Delta) + \sigma^2_{k|k-1} + 1}[dN(k\Delta) - \lambda(k\Delta)] \]  

\[ \sigma^2_{k|k} = [\sigma^2_{k|k-1}]^{-1} + \lambda(k\Delta) \]  

(4.3)

(4.4)

The variance updating algorithm in the Fahrmeir algorithm agrees because the observed and expected Fisher information are the same for the Poisson model in our example. The state updating equation differs from our
updating formula in equation 2.15 because their update is computed from
the Kalman filter and not directly by finding the root of the log posterior
probability density. The state and variance update formulae in the Durbin
and Koopman algorithm differ from ours because theirs use a Taylor series
approximation of the score function, first derivative of the log likelihood,
instead of the exact score function. Fahrmeir (1992) and Fahrmeir and Tutz
(1994) suggest using the EM algorithm for estimating the unknown param-
eters, but details are not given.

In an example applied to spike train data, Sahini (1999) describes the
use of a latent model for neural firing where spikes are generated as an
inhomogeneous Polya process. In his model, parameters are computed by
optimizing the marginalized posterior by gradient ascent, and Monte Carlo
goodness-of-fit is used to compare the model fit with measured spike train
stochastic process.

The fact that our state-space models fit the simulated data better than
the empirical method is expected given that the spikes were generated with
the model. In applications to real data, it will be possible to use the same
approach, testing reasonable and ideally parsimonious forms of the state-
space and point process models for a given neurophysiological experiment.
In any case, we may use the time-rescaling theorem to assess goodness-of-fit
of any candidate models.

To study the problem of estimating a latent process simultaneously with
its model parameters and the parameters of the observation process, we
discretized time and assumed that the observation process and the latent
process occur on a lattice of points spaced \( \Delta_1 \) time units apart. Using
the EM algorithm, we computed the maximum likelihood estimate of \( \theta \)
and empirical Bayes’ estimates of the latent process conditional on the maximum
likelihood estimate of \( \theta \). A Bayesian alternative would be to specify a prior
distribution for \( \theta \) and compute a joint posterior probability density for the
latent process and \( \theta \). Liu and Chen (1998) developed sequential Monte Carlo
algorithms that may be adapted to this approach.

As another alternative, it is useful to point out how the latent process and
parameter estimation may be carried out if both the point process and
the latent process are assumed to be measured in continuous time. Nonlin-
ear continuous time filtering and smoothing algorithms for point process
observations have been studied extensively in the control theory literature
(Snyder, 1975; Segall, Davis, & Kailath, 1975; Boel & Beneš, 1980; Snyder
& Miller, 1991; Twum-Danso, 1997; Solo, 2000; Twum-Danso & Brockett,
2001). If the normalized conditional probability density, \( p(x(t) \mid N_{0,t}) \), is
to be evaluated in continuous time, then a nonlinear stochastic partial dif-
ferential equation in this probability density must be solved at each step
(Snyder, 1975; Snyder and Miller, 1991). Here we let \( x(t) \) be the continuous
time value of \( x_k \). If the updating is performed with respect to the unnorma-
lized conditional probability density, \( p(x(t), N_{0,t}) \), then a linear stochastic
partial differential equation must be solved in this probability density at
each step (Boel & Beneš, 1980; Twum-Danso, 1997; Solo, 2000). For either
the normalized or unnormalized probability density updating algorithms,
the essential steps in their derivations use the posterior prediction equation
in equation A.1 and the one-step prediction equation in equation A.2 to de-
rive Fokker-Planck equations (Snyder, 1975; Snyder & Miller, 1991; Twum-
Danso, 1997; Solo, 2000; Twum-Danso & Brockett, 2001). If the parameters
of the continuous time system are nondynamic and unknown, then, as in
the discretized time case we present here, either the normalized or unnor-
malized partial differential equation updating of the conditional probability
density may be embedded in an EM algorithm to compute maximum likeli-
hood estimates of the parameters and empirical Bayes estimates of the latent
process. Similarly, a Bayesian procedure can be derived if a prior probabil-
ity density for $\theta$ can be specified. While the normalized and unnormalized
conditional probability density updating equations have been known for
several years, the computational requirements of these algorithms may be
the reason they have not been more widely used (Manton, Krishnamurthy,
& Elliott, 1999). Discretized approximations (Snyder & Miller, 1991; Twum-
Danso, 1997; Twum-Danso & Brockett, 2001) and sequential Monte Carlo
algorithms (Solo, 2000; Doucet, de Freitas, & Gordon, 2001; Shoham, 2001)
have been suggested as more plausible alternatives. The sequential Monte
Carlo methods use simulations to compute recursively the solutions to equa-
tions A.1 and A.2 on a discrete lattice of time points, whereas our nonlinear
recursive algorithm, equations 2.12 through 2.16, uses sequential gaussian
approximations to perform the same computations.

A potential application of this analysis paradigm would be to estimate
the effect of external cue on a spike train in the delayed-response hand-
pointing task described in Riehle et al. (1997). In this experiment, parallel
spike data, measured in primary motor cortex of the monkey, are analyzed
to estimate differences between spike rate increases corresponding to ac-
tual motion and those caused by expectation of a stimulus. The structure
of our model enables us to estimate above-random firing propensity in a
single cell while incorporating the history of cell firing. For these reasons,
this approach may have advantages over the unitary events analysis meth-
ods (Grun et al., 1999), which may be difficult to apply to neurons with
low firing rates (Roy, Steinmetz, & Niebur, 2000). A second potential
application would be in the analysis of cell firing in the trace conditioning
paradigm (McEchron, Weible, & Disterhof, 2001). In this case, a conditioned
stimulus is followed after a trace interval by an unconditioned stimulus.
After many such trials, an association between the two stimuli develops,
as evidenced by changes in the firing rate of the neuron. These data are
conventionally analyzed using PSTH techniques. However, because these
studies involve an implicit relation between the stimulus and the neural
spiking activity, this relation may be more clearly delineated by using the
paradigm presented here. Finally, we are currently investigating the appli-
cation of these methods to learning and memory experiments during
recordings from the medial temporal lobe of the monkey (Wirth et al., 2002; Yanike et al., 2002).

The state-space approach suggests several advantages. First, the approach uses a latent variable to relate explicitly the timing of the stimulus input and history of the experiment to the observed spiking activity. Second, use of explicit probability models makes it possible to compute probability density functions and confidence intervals for model quantities of interest. Finally, formulation of the analysis in terms of a general point process model allows us to assess model goodness-of-fit using the K-S tests based on the time-rescaling theorem. In our opinion, this latter step is the most critical as it forces us to assess how sensitive our inferences may be to lack of agreement between the model and the experimental data.

In summary, we have presented a computationally tractable method for state-space and parameter estimation from point process observations and suggested that these algorithms may be useful for analyzing neurophysiologic experiments involving implicit stimuli. In a future publication, we will apply these methods to actual experimental studies.

Appendix: Derivation of the Recursive Nonlinear Filter Algorithm

We derive a form of the recursive filter equations appropriate for an arbitrary point process model. The algorithm in equations 2.12 through 2.16 is obtained by taking the special case of the Poisson. To derive the nonlinear recursive filter, we require the posterior prediction equation,

\[ p(x_k | H_k) = \frac{p(x_k | H_{k-1})p(dN(k\Delta) \mid x_k, H_k)}{p(dN(k\Delta) \mid H_{k-1})}, \tag{A.1} \]

and the one-step prediction or Chapman-Kolmogorov equation,

\[ p(x_k | H_{k-1}) = \int p(x_k \mid x_{k-1})p(x_{k-1} \mid H_{k-1}) \, dx_{k-1}. \tag{A.2} \]

The derivation of the algorithm proceeds as follows. Assume that at time \((k-1)\Delta\), \(x_{k-1} \mid k-1\) and \(\sigma^2_{k-1} \mid k-1\) are given. Under a gaussian continuity assumption on \(x_k\), the distribution of \(x_k\) given \(x_{k-1} \mid k-1\) is \(N(\rho x_{k-1} \mid k-1 + a_k, \sigma^2_{k-1} \mid k-1)\), where \(\sigma^2_{k-1} \mid k-1 = \sigma^2 + \rho^2 \sigma^2_{k-1} \mid k-1\). By equations 2.4, A.1, and A.2 and the gaussian continuity assumption, the posterior probability density \(p(x_k \mid H_k)\) and the log posterior probability density \(\log p(x_k \mid H_k)\) are, respectively,

\[ p(x_k \mid H_k) \propto \exp \left\{ -\frac{1}{2} \frac{(x_k - \rho x_{k-1} \mid k-1 - a_k)^2}{\sigma^2_{k-1} \mid k-1} \right\} \]

\[ \times \prod_{c=1}^{C} \exp \left\{ \log \lambda_c(k\Delta \mid H_c^k) dN^c(k\Delta) - \lambda_c(k\Delta \mid H_c^k) \Delta \right\} \tag{A.3} \]
\begin{equation}
\log p(x_k \mid H_k) \propto -\frac{1}{2} \frac{(x_k - \rho x_{k-1|k-1} - \alpha I_k)^2}{\sigma^2_{k|k-1}} + \sum_{c=1}^{C} \left[ \log \lambda_c(\Delta \mid H_c) dN^c(\Delta) - \lambda_c(\Delta \mid H_c) \Delta \right]. \quad (A.4)
\end{equation}

To find the optimal estimate of \( x_k \), we apply a gaussian approximation to equation A.3. This gaussian approximation is distinct from the gaussian continuity assumption and means that we can use the mode and variance of the probability density in equation A.3 to approximate it as a gaussian probability density. As a result, we differentiate with respect to \( x_k \) to find the mode, and we compute the second derivative to obtain the approximate variance (Tanner, 1996). Differentiating equation A.4 with respect to \( x_k \) gives

\begin{equation}
\frac{\partial \log p(x_k \mid H_k)}{\partial x_k} = -\frac{(x_k - \rho x_{k-1|k-1} - \alpha I_k)}{\sigma^2_{k|k-1}} + \sum_{c=1}^{C} \frac{1}{\lambda_c(\Delta)} \frac{\partial \lambda_c(\Delta \mid H_c)}{\partial x_k} [dN^c(\Delta) - \lambda_c(\Delta \mid H_c) \Delta]. \quad (A.5)
\end{equation}

and solving for \( x_k \) yields

\begin{equation}
x_k = \rho x_{k-1|k-1} + \alpha I_k \\
+ \sum_{c=1}^{C} \sigma^2_{k|k-1} \frac{\partial \lambda_c(\Delta \mid H_c)}{\partial x_k}^{-1} \\
\times \frac{\partial \lambda_c(\Delta \mid H_c)}{\partial x_k} [dN^c(\Delta) - \lambda_c(\Delta \mid H_c) \Delta]. \quad (A.6)
\end{equation}

Equation A.6 is in general nonlinear in \( x_k \) and can be solved using Newton’s method. The second derivative of equation A.5 is

\begin{equation}
\frac{\partial^2 \log p(x_k \mid H_k)}{\partial x_k^2} = -\frac{1}{\sigma^2_{k|k-1}} + \sum_{c=1}^{C} \left[ \left( \frac{\partial^2 \lambda_c(\Delta \mid H_c)}{\partial x_k^2} - \frac{1}{\lambda_c(\Delta)} \left( \frac{\partial \lambda_c(\Delta \mid H_c)}{\partial x_k} \right)^2 \frac{1}{\lambda_c(\Delta)^2} \right) \right. \\
\left. \times [dN^c(\Delta) - \lambda_c(\Delta \mid H_c) \Delta] \\
- \left( \frac{\partial \lambda_c(\Delta \mid H_c)}{\partial x_k} \right)^2 \frac{1}{\lambda_c(\Delta)^2} \Delta \right], \quad (A.7)
\end{equation}
and the variance of $x_k$, under the gaussian approximation to equation A.3, is

$$\sigma^2_{k|k} = -\left[ -\frac{1}{\sigma^2_{k|k-1}} + \sum_{c=1}^{C} \left( \frac{\partial^2 \lambda_c(k\Delta)}{\partial x_k^2} \frac{1}{\lambda_c(k\Delta)} - \left( \frac{\partial \lambda_c(k\Delta)}{\partial x_k} \right)^2 \frac{1}{\lambda_c(k\Delta)^2} \right) \right. \times \left[ dN_c(k\Delta) - \lambda_c(k\Delta) \Delta \right]$$

$$\left. - \left( \frac{\partial \lambda_c(k\Delta)}{\partial x_k} \right)^2 \frac{1}{\lambda_c(k\Delta)^2} \right]^{-1}. \quad (A.8)$$

Equations A.6 and A.8 constitute the basis for the general form of the filter equations 2.12 through 2.16.

Acknowledgments

Support was provided in part by NIMH grants MH59733, MH61637, NIDA grant DA015644, NHLBI grant HL07901, and NSF grant IBN-0081458. Part of this research was performed while E.N.B. was on sabbatical at the Laboratory for Information and Decision Systems in the Department of Electrical Engineering and Computer Science at MIT.

References


Received April 24, 2002; accepted November 18, 2002.