J Comput Neurosci (2015) 38:499–519
DOI 10.1007/s10827-015-0551-y

Fast maximum likelihood estimation using continuous-time neural point process models

Kyle Q. Lepage · Christopher J. MacDonald

Received: 30 May 2014 / Revised: 5 January 2015 / Accepted: 2 March 2015 / Published online: 20 March 2015

Abstract A recent report estimates that the number of simultaneously recorded neurons is growing exponentially. A commonly employed statistical paradigm using discrete-time point process models of neural activity involves the computation of a maximum-likelihood estimate. The time to computate this estimate, per neuron, is proportional to the number of bins in a finely spaced discretization of time. By using continuous-time models of neural activity and the optimally efficient Gaussian quadrature, memory requirements and computation times are dramatically decreased in the commonly encountered situation where the number of parameters p is much less than the number of time-bins n. In this regime, with q equal to the quadrature order, memory requirements are decreased from $O(np)$ to $O(qp)$, and the number of floating-point operations are decreased from $O(np^2)$ to $O(qp^2)$. Accuracy of the proposed estimates is assessed based upon physiological consideration, error bounds, and mathematical results describing the relation between numerical integration error and numerical error affecting both parameter estimates and the observed Fisher information. A check is provided which is used to adapt the order of numerical integration. The procedure is verified in simulation and for hippocampal recordings. It is found that in 95 % of hippocampal recordings a q of 60 yields numerical error negligible with respect to parameter estimate standard error. Statistical inference using the proposed methodology is a fast and convenient alternative to statistical inference performed using a discrete-time point process model of neural activity. It enables the employment of the statistical methodology available with discrete-time inference, but is faster, uses less memory, and avoids any error due to discretization.

Keywords Neural spiking · Statistical analysis · Fast · Point process · Estimation · Continuous-time · Gaussian quadrature

1 Introduction

A recent report estimates the number of recorded neurons to be growing exponentially (Stevenson and Kording 2011). Exponential growth of data presents an increasingly important challenge to the effective application of contemporary statistical methods in the neurosciences, where computation times are beginning to exceed acceptable limits. The growing challenge is being met by recent developments within the neuroscience community. In Paninski et al. (2010) state-space models, Markov-chain Monte Carlo methods of neural decoding, and spatially varying firing rates are estimated in O(n) time, enabled by the banded structure of relevant matrices. Here n is the number of time-steps in the observation duration. In Citi et al. (2014), computational efficiency is gained be replacing a Riemann approximation with an improved, but discrete, approximation to an integral appropriate for restricted but often realistic models of neural spiking activity.

In Ramirez and Paninski (2013) maximum likelihood estimates (MLE) computed for generalized linear models are hastened by replacing estimation with the log-likelihood

Action Editor: Liam Paninski

K. Q. Lepage (✉)
Department of Mathematics & Statistics, Boston University, Boston, MA 02215, USA
e-mail: lepage@math.bu.edu

C. J. MacDonald
Picower Institute for Learning and Memory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

with estimation using the expectation with respect to stochastic covariates of a stochastic log-likelihood. Here focus is placed upon the computational cost arising from the log-likelihood term that is nonlinear in the model parameters.

Similar to this latter work, in the proposed methodology focus is placed upon a computationally efficient approximation to an analagous nonlinear term in the log-likelihood of continuous-time point processes possessing an exponential conditional intensity model. This procedure uses a non-stochastic log-likelihood, consequently requiring no assumptions regarding a probabilistic description of model covariates. The analagous nonlinear term in the proposed methodology requires the integration of the rate, or conditional intensity, over the observation interval. By employing the optimal Gaussian quadrature rule to approximate this integral the number of evaluations of the conditional intensity when computing this integral reduces from n in the standard discrete-time situation (Truccolo et al. 2005), to q where $q \ll n$, ultimately reducing an $O(np^2)$ computational cost to $O(qp^2)$.[1] This computational complexity compares favorably with that of the discrete-time MLE counterparts studied in, for e.g. (Truccolo et al. 2005; Paninski 2004) which have a computational complexity of $O(np^2)$.

Gaussian quadrature is known to be much faster than competitive methods in low-dimensional settings.[2] It has been known to the statistics community for a very long time, but has recently been largely ignored; possibly for three main reasons: (i) there is a community focus upon Markov Chain Monte-Carlo techniques in favor of classical quadrature, (ii) there is a current focus upon numerical integration in higher dimensions, and (iii) the standard Gaussian quadrature scheme is not adaptive – it is difficult to assess accuracy despite an expression for the numerical integration error (Kuonen 2003).

In Genz and Kass (1991),Genz and Kass (1997) methods based upon comparing successive Gaussian quadrature approximations of increasing orders are discussed. These methods are found to be promising but conservative, in the sense that the quadrature order in this scheme is often specified to be unnecessarily large. In Genz and Kass (1997) a method of adaptively determining the error as a linear combination of a theoretical error term with the difference between successively increasing orders of approximation is presented. In this scheme, the order of integration is increased until the error falls below a user defined threshold.

[1] Gaussian quadrature is a method of numerical integration first developed by Gauss. It is optimal for integrating polynomials (Davis and Rabinowitz 1967).

[2] Except for Clenshaw-Curtis quadrature. This quadrature scheme is a factor of 2 less computationally efficient at low quadrature orders q, and becomes computationally as efficient as q increases (Trefethen 2008).

In the preparation of this manuscript, other work using Gaussian quadrature for the estimation of refractory neural point-process models was made known to the authors (Mena and Paninski 2014).

In this work, based upon a physiological consideration bounding quadrature error, an empirical approximate error bound, and a theoretical expression linking quadrature error to parameter estimate error, quadrature order is increased until the parameter estimate error is small relative to either a physiologically meaningful scale or to parameter standard error. Despite often over-estimating the order of integration, up to an order of magnitude improvement in computation time can be realized for the setting $n \gg p$.

The paper begins in Section 2.1 with a review of maximum-likelihood estimation (MLE) for a continuous-time point process with an exponential model of the conditional intensity. In Section 2.2 background material on Gaussian quadrature is presented and in Section 2.3 the proposed approximate MLE for continuous-time point process models with exponential conditional intensities is presented. The computational and memory requirments needed to compute discrete-time and continuous-time MLEs is provided in Sections 2.4, 2.5 and 3. The effect of quadrature inaccuracy is quantified in Section 4, while the accuracy of Gaussian quadrature for computing the integral of a physiologically plausible conditional intensity is discussed in Section 5. The full proposed adaptive procedure is presented in Section 6. The proposed MLE is simulated in Section 7 on a synthetic time-cell, and the proposed procedure is applied to recordings of actual hippocampal time-cells in Section 8. The paper concludes with a discussion in Section 9.

2 Background

2.1 Continuous-time point process MLE

Let $N(t)$ be an ordered, continuous-time counting process with a conditional intensity, λ, conditioned on time-dependent covariates x_t, the process history H_t, the vector-valued parameter β, and history kernel, γ:

$$\lambda(t|\,\beta,\gamma,x_t,H_t) = \lim_{\delta\to 0} P\left(N(t+\delta)-N(t)>0 \mid \beta,\gamma,x_t,\mathrm{H}_t\right)\big/\delta \ , \qquad (1)$$

with

$$log\left(\lambda(t|\beta,\gamma,x_t,H_t)\right) = \sum_{k=1}^{p}\beta_k x_k(t) + \int_{-\infty}^{t}\gamma(t')\,dN(t-t') \ . \qquad (2)$$

Here $dN(t) = N(t+\delta) - N(t)$ is an infinitesimal increment of the counting process $N(t)$ and p is the number of parameters. Thus, the conditional intensities considered

in this work possess two multiplicatively separable components. The first component, λ_p is Poisson in the sense that it does not depend upon past neural spiking, and the second component, λ_r, captures history effects. Specifically,

$$log\left(\lambda(t|\,\beta,\gamma,x_t,H_t)\right) = log\left(\lambda_p(t|\beta,x_t)\right) + log\left(\lambda_r(t|\gamma,H_t)\right). \quad (3)$$

For the case where λ_r is equal to zero, the increment process dN is Poisson. In the general case, $\lambda_r, \lambda_p \neq 0$, dN is a point process (Daley and Vere-Jones 2003; Snyder 1975).

The parameterization of λ based upon β, γ, x_t and history effects H_t possesses as special cases the class of generalized linear models, and generalized additive models. Common models of $log(\lambda_p)$ include polynomials and splines as a function of, for e.g., space and/or time, and with parameters that depend upon binary variables representing experimental condition. Similarly, history dependence captured by $log(\lambda_r)$, is often an integral or sum over past spiking activity, which can be usefully parameterized in a multitude of ways. For notational convenience it is useful to represent, $\lambda(t|\beta,\gamma,x_t,H_t)$ as $\lambda(t|\beta)$.

Given the observation of a time-series of event occurrences, the log-likelihood with respect to β is Daley and Vere-Jones (2003),Snyder (1975)

$$\ell(\beta) = \sum_{t\in T_s} \log\left(\lambda(t|\beta)\right) - \int_0^T \lambda(t'|\beta)\,dt', \quad (4)$$

for T_s equal to the set of times at which the neuron activated. The duration of observation is T. The maximization of ℓ with respect to the unknown parameter β requires, for typical scenarios, the numerical computation of an approximation to the integral J:

$$J(\beta) = \int_0^T \lambda(t'|\beta)\,dt'. \quad (5)$$

One such approximation is computed with Riemann quadrature:

$$J(\beta) \approx \Delta \sum_{j=0}^{n} \lambda\left(j\,\Delta\,|\,\beta\right), \quad = J_n^{(r)}(\beta). \quad (6)$$

Here Δ is the step size in seconds and $n = \lfloor T/\Delta \rfloor$. This approximation leads to an expression for the log-likelihood that is identical to those used in the discrete-time case, as in Truccolo et al. (2005). In this case,

$$\left|\int_0^T \lambda(t'|\beta)dt' - J_n^{(r)}(\beta)\right| \leq \sup_{\zeta\in(0,T)} \frac{n\Delta^2}{2}\left|\lambda^{(1)}(\zeta|\beta)\right|, \quad (7)$$

for λ differentiable on $(0,T)$. Because $\lambda(t|\beta)$ can vary rapidly in time due to, for example, refractory effects, it is often necessary to work with $\Delta \ll T$. A popular choice of Δ in standard neuroscience applications is 1 ms, a value specified to capture the refractory period exhibited by a neuron immediately following an action potential. Because T can be large relative to Δ, n can be large, and the computational cost of Eq. 6 is the $n+1$ evaluations of the integrand λ. This cost propagates through the commonly employed algorithms used to compute maximum-likelihood estimates and results in a $O(np^2)$ computational cost. In this paper, by reducing n to q with q near 50, a substantial reduction in computational cost results.

2.2 Gaussian quadrature

Gaussian quadrature is a classical quadrature rule due to Gauss, and there are many treatments of the topic. Books covering Gaussian quadrature rules are listed in the beginning of Trefethen (2008). Briefly, the Gaussian quadrature rule J_q approximates the integral J as

$$J_q(\beta) = \sum_{j=1}^{q} w_j\,\lambda(t_j|\beta). \quad (8)$$

Here t_j is chosen as the roots of the q^{th} order Legendre polynomial, $L_q(t)$. The weights w_j satisfy orthogonality properties described in Appendix A. A basic derivation, ignoring questions of existence and uniqueness discussed in, for e.g., (Stoer and Bulirsch 2002), is see also provided in Appendix A. Gaussian quadrature integrates order $2q-1$ polynomials exactly and has an error:

$$\int_0^T \lambda(t|\beta)\,dt - \sum_{j=1}^{q} w_j\lambda(t_j|\beta) = \frac{\lambda^{(2q)}(\zeta|\beta)}{(2q)!\,k_q^2}, \quad (9)$$

for $2q$-differentiable λ on the interval $(0,T)$ and for some $\zeta \in (0,T)$ (Davis and Rabinowitz 1967, p. 75). Here k_q is the lead coefficient in the q^{th} order Legendre polynomial:

$$L_q(t) = k_q \prod_{i=1}^{q}(t-t_i), \quad (10)$$

with t_i the i^{th} root of L_q.

While appealing, the quadrature accuracy and hence the order of integration q required for accurate inference are apriori unknown. Further, while classically known to be computationally more efficient than competing quadrature rules, Gaussian quadrature does not permit the re-use of quantities computed from the lower quadrature order computations, discussed for e.g. in Stoer and Bulirsch (2002). This reduces the appeal of adaptive strategies employing successively higher quadrature orders until result stabilization.

In the current setting of continuous-time maximum likelihood estimation of a parameterized conditional intensity another adaptive strategy is proposed. This strategy provides estimates in computation times up to an order of magnitude faster than current algorithms. This decreased computation time is accompanied by the increase in accuracy afforded by continuous-time point process models of neural activity.

The time-discretization used to compute the integral in Eq. (4) using Gaussian quadrature suggests a novel discrete-time model of neural activity based upon the Gaussian quadrature nodes, t_j, $j = 1, \ldots, q$. This discretization leads to a non-orderly random process model with a log-likelihood that may differ from Eq. (4) (see Appendix B). In this work, focus is placed upon continuous-time point process models of neural spiking. For clarity, in the following a discrete-time model refers to the discrete-time point process model of neural activity described in Section 2.1 and in Truccolo et al. (2005). For this model, Δ is a constant value independent of time.

2.3 Approximate MLE $\hat{\beta}_q$ (using gaussian quadrature)

Associated with the q^{th} order Gaussian quadrature, J_q, is the q^{th} order approximation ℓ_q of the log-likelihood ℓ:

$$\ell_q(\beta) = \sum_{t \in T_s} \log\left(\lambda(t|\beta)\right) - \sum_{j=1}^{q} w_j \lambda(t_j|\beta) \,, \tag{11}$$

the q^{th} order approximation, $\mathbf{s}_q$, to the score equations $\mathbf{s}$:

$$\begin{aligned} \mathbf{s}_q(\beta) &= \nabla_\beta \ell_q(\beta) \,, \\ &= \sum_{t \in T_s} \frac{\nabla_\beta \lambda\,(t|\beta)}{\lambda\,(t|\beta)} - \sum_{j=1}^{q} w_j \nabla_\beta \lambda(t_j|\beta) \,, \end{aligned} \tag{12}$$

and the q^{th} order approximation, $\mathbf{I}_q$, to the Hessian, $\mathbf{I}$:

$$\begin{aligned} \mathbf{I}_q(\beta) &= \nabla^2_\beta \ell_q(\beta) \,, \\ &= -\sum_{j=1}^{q} w_j \nabla^2_\beta \lambda(t_j|\beta) \,. \end{aligned} \tag{13}$$

Note that $\mathbf{I}_q$ evaluated at row r and column c is

$$\left[\mathbf{I}_q(\beta)\right]_{r,c} = -\sum_{j=1}^{q} w_j \; x_{t_j,a} \; x_{t_j,b} \; e^{\sum_{j'=1}^{p} x_{t_j,j'} \beta_{j'}} \,. \tag{14}$$

Due to the concavity of ℓ_q (established in Appendix E), the approximate MLE $\hat{\beta}_q$ uniquely satisfies:

$$\mathbf{s}_q(\hat{\beta}_q) = 0 \,. \tag{15}$$

2.4 Computation of $\hat{\beta}_q$

In this work Newton-Raphson iteration is used to compute $\hat{\beta}_q$, the approximate maximum likelihood estimate of β for the continuous-time model of neural activity specified in Eq. (2).

Newton-Raphson iteration begins with an initial guess $\hat{\beta}^{(0)}$ of the approximate maximum likelihood estimate $\hat{\beta}_q$. Subsequent estimates are computed according to:

$$\hat{\beta}^{(i)} = \hat{\beta}^{(i-1)} - \left[\mathbf{I}_q\left(\hat{\beta}^{(i-1)}\right)\right]^{-1} \mathbf{s}_q\left(\hat{\beta}^{(i-1)}\right) \,. \tag{16}$$

The number of floating point operations (FLOPs) required to evaluate a single Newton-Raphson iteration, along with related quantities, is provided in Table 1. The update specified in Eq. (16) can be computed using QR decomposition, and for standard neural firing rates requires $O(qp^2)$ floating

Table 1 Key Quantities: Required Floating Point Operations (FLOPs)

Quantity	FLOPs	Equation
$\lambda(t\|\beta)$	p multiplications, $p-1$ additions, 1 exp().	Eq. (2)
$\nabla_\beta \lambda(t\|\beta)$	As for $\lambda(t\|\beta)$ plus p multiplications.	Term 2 of Eq. (12)
$\nabla_\beta \lambda(t\|\beta) \,/\, \lambda(t\|\beta)$	Zero due to cancellation.	Term 1 of Eq. (12)
$\sum_{t \in T_s} \log \lambda(t\|\beta)$	$\|T_s\|$ evaluations of $\log \lambda(t\|\beta)$,	
	plus $\|T_s\|$ additions.	Term 1 of Eq. (12)
$\sum_{j=1}^{q} w_j \lambda(t_j\|\beta)$	q evaluations of $\lambda(t\|\beta)$, plus q multiplications,	
	plus $q-1$ additions.	Term 2 of Eq. (12)
$\mathbf{s}_q(\beta)$	Term 1 FLOPS plus Term 2 FLOPS (of Eq. (12))	Eq. (12)
$\mathbf{I}_q(\beta)$	$q\frac{p^2+p}{2}\times$ (an evaluation of $\lambda(t\|\beta)$ $plus$ 3 multiplications)	Eq. (14)
$\mathbf{I}_q^{-1}(\beta)$	$O(p^3)$ in general.	Eq. (16)
$\hat{\beta}^{(i)}$	$O(qp^2)$ using QR decomposition.	Eq. (16)

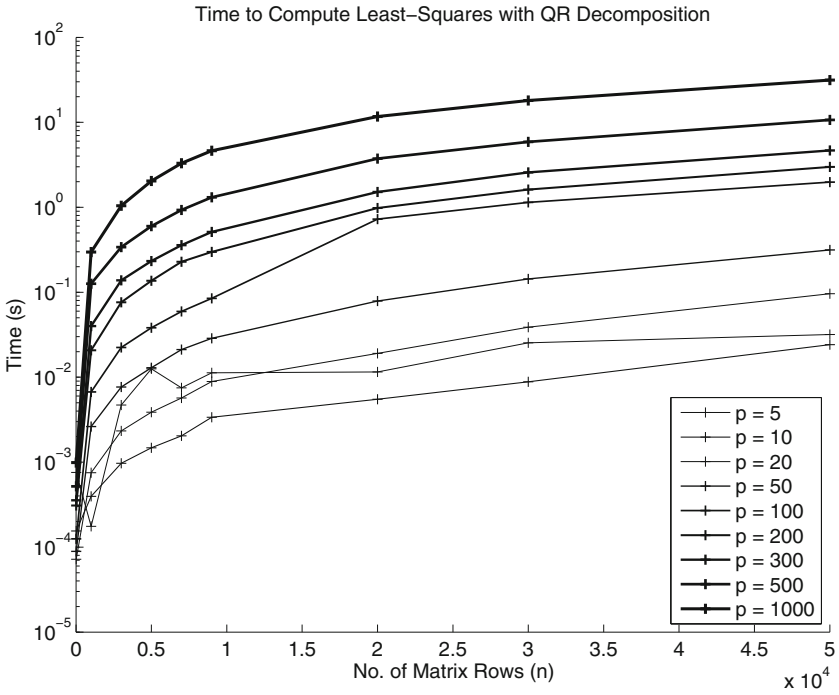

Fig. 1 Time to compute least-squares via the fast QR-decomposition is $O(np^2)$. Each point corresponds to a single time-to-compute, n, p triplet. These triplets are plotted with respect to varying n (the number of rows), with each black curve corresponding to a different value of p (the number of parameters)

point operations to compute Fig. 1.[3,4] The required number of FLOPs required for a discrete-time Newton-Raphson update is obtained by replacing q with n. In this case the number of FLOPs is $O(np^2)$.

2.5 Iterated re-weighted least squares (IRLS)

IRLS is an alternative iterative algorithm for computing the maximum likelihood estimate for a discrete-time point process model of neural activity. It commonly involves a QR decomposition that requires $O(np^2)$ floating point operations (see Fig. 1). Thus in the discrete-time case, whether using Newton-Raphson iteration or IRLS, the computational cost is $O(np^2)$, for $n > p$. The analagous IRLS algorithm to compute continuous-time MLEs has not been reported. In this work maximum likelihood estimates of discrete-time point process model parameters are computed using IRLS, using the MATLAB function glmfit().

3 Memory requirements

Time is required to form and manipulate large matrices. The type of memory available for storage while performing computations can influence the speed of computation. Typically fast memory, such as central-processing unit (CPU) cache and CPU registers are more limited than slower types of memory such as more standard random access memory (RAM) and non-volatile hard-disk storage. When designing algorithms that efficiently use parallel processing units, communication latencies between processors is an issue. The less numbers that must be communicated the better, and typically algorithms that use less memory need to communicate fewer numbers. The proposed methodology makes more efficient use of memory.

Standard implementations of the Newton-Raphson algorithm and the iterative re-weighted least squares algorithm make use of an intermediary model matrix. This model matrix is n x p when using Riemann quadrature and it is q x p when using Guassian quadrature. For 100 units, 10 parameters (p is 10) and a ten second recording discretized to 1 ms time-bins so that n is 10000, requiring approximately 10^8 megabytes when using double floating point precision. For the same data, a q equal to 60 may suffice (see Section 8), reducing this memory requirement by more than a factor of one hundred.

[3]This is assuming that the Hessian has no special structure. When the Hessian has special structure the required FLOPS to compute $\hat{\beta}^{(i)}$ can, depending on the specific nature of the structure, be reduced from $O(qp^2)$. The structure of the Hessian depends upon the model of the conditional intensity and cannot, in general, be guaranteed. Thus, while there may exist interesting circumstances where the required $O(qp^2)$ FLOPS is reduced, emphasis in this work is on the more general setting.

[4]Other fast methods of iteratively updating $\hat{\beta}^{(i)}$ exist that can be effective (Shewchuk 1994). For e.g. in Shewchuk (1994), conjugate gradient iteration is reported to require $O(m\sqrt{\kappa})$ operations for solving the problem, $\mathbf{Ax} = \mathbf{b}$. Here the matrix $\mathbf{A}$ possesses m non-zero entries and has a condition number κ. Note that m in the context considered in this work, is equal to qp for the continuous-time case, and np for the discrete-time case. Thus, without extra assumptions conjugate gradient also requires either $O(qp^2)$ or $O(qn^2)$ FLOPS.

4 Accuracy of $\hat{\beta}_q$

The accuracy of J_q is discussed in Section 5, a computational check for the accuracy of $\hat{\beta}_q$ is provided in Steps 2-8 of the complete algorithm presented in Section 6. Here focus is placed upon the accuracy of the approximate MLE $\hat{\beta}_q$ as J_q nears the actual integral of the conditional intensity J.

As J_q approaches J, one expects the approximate MLE $\hat{\beta}_q$ to approach the actual MLE $\hat{\beta}$. This intuitive result is proven in a short sequence of lemmas provided in Appendices E and F. The main result, Eq. (89), is reproduced here:

$$\left|\hat{\beta} - \hat{\beta}_q\right| \leq 2 \sqrt{\frac{\left|\lambda^{(2q)}(\zeta \mid \hat{\beta}_q)\right|}{(2q)!\, k_q^2 \left|\frac{d^2 \ell_q(\hat{\beta}_q)}{d\beta^2}\right|}}. \tag{17}$$

When the quadrature error $|\ell_q(\hat{\beta}_q) - \ell(\hat{\beta})|$ is small, and the approximate observed Fisher information, $-\ell_q''(\hat{\beta}_q)$ is large, the separation between $\hat{\beta}_q$ – the MLE computed from $\ell_q(\beta)$ – and the actual MLE $\hat{\beta}$ is small. The approximate observed Fisher information, $-\ell_q''(\hat{\beta}_q)$, is large for positive rate and sufficiently large observation intervals. These conditions are typically encountered when analyzing data collected in common neuroscience experiments.

For the purposes of performing inference with the standard maximum-likelihood methodology presented in Truccolo et al. (2005), it is required that $\hat{\beta}_q$ possess the sampling properties of the actual maximum likelihood estimates computed from the exact log-likelihood, ℓ. These asymptotic properties – valid under mild conditions when the duration of observation grows large – are often used for the purposes of assessing statistical significance and performing statistical inference. In Appendix G these properties are established.

5 Accuracy of J_q

In Section 4 and Appendix F the accuracy of the approximate MLE $\hat{\beta}_q$ is established when J_q is an accurate approximation of J. The accuracy of J_q depends only on the higher-order derivatives of the statistical model of the conditional intensity. In the application of the methodology presented in this work, the accuracy of J_q for a specified conditional intensity model is implicitly established using an algorithm presented in Section 6 to estimate the effect of quadrature error upon the maximum likelihood parameter estimate, $\hat{\beta}$.

The utility of the algorithm presented in Section 6 depends upon the existence of a reasonable q such that J_q is close to J. The existence of such a reasonable order is discussed in Section 5.1 for exponentiated polynomial models of the conditional intensity, and in Section 5.2 for the exponentiated spline models. It is found that these models include models with infinite order polynomials and do not provide a guarantee for the existence of a useable quadrature order q. The models discussed in Sections 5.1 & 5.2 ignore history effects; however, they can be multiplicatively augmented to incorporate history effects.

In Section 5.3 it is found that for a plausible model of the neural refractory event a quadrature order of 20 is sufficient. This result implies (i) that a 39^{th} order polynomial is sufficient to accurately model what is arguably the most dramatic change in the probability of an action potential that a neuron will undergo and (ii) that 39^{th} order polynomials can capture very dramatic changes in conditional intensity. Results (i) and (ii) suggest that Guassian quadrature will accurately integrate adequately complicated intensity models for $q \ll n$.

5.1 Exponential models & polynomials

Commonly exponentiated polynomials are used to model the conditional intensity, in this case the rate of an inhomogeneous Poisson process, often as splines, for e.g. (MacDonald et al. 2011; Barbieri et al. 2004; Kass et al. 2003). Thus,

$$\lambda(t|\beta) = \exp\left(\sum_{j=1}^{p} \beta_j\, z_j(t)\right), \tag{18}$$

in a standard model. Here z_j is a polynomial of order j. The example model Eq. (18) can be written as

$$\lambda(t|\beta) = \sum_{k=0}^{\infty} \frac{\left(\sum_{j=1}^{p} \beta_j z_j(t)\right)^k}{k!}. \tag{19}$$

Thus the exponential model is an infinite order polynomial. This polynomial may be approximated by a reduced-order polynomial with an accuracy that depends upon β_j, $j = 1, \ldots, p$.

5.2 Spline models

Spline models are piecewise-connected polynomials that satisfy continuity constraints across times of connection (knots). At the knots the higher-order derivatives of the exponentiated spline may not exist, and the assumptions leading to the error Eq. (9) do not hold. Since the spline model is continuous, these difficulties are avoided by noting that there exists a polynomial that uniformly

approximates the spline (Weirstrass representation theorem, for β restricted to a plausible, closed interval). With this connection, the discussion in Section 5.1 applies to the spline model without modification.

5.3 The neural refractory effect

The neural refractory effect is the process where a neuron undergoing an action potential ceases for a duration to be able to initiate a subsequent action potential. Its onset is arguably the most rapid change in the probability of spiking that can occur. For a continuous-time point process model of neural activity, this imposes an upper-bound on the absolute rate-of-change of the conditional intensity.

Note that the absolute difference between J and J_q can be upper-bounded:

$$\begin{aligned}\left| \int_a^b \lambda(t|\beta)\, dt - \sum_{j=1}^{q} w_j \lambda(t_j|\beta) \right| = \\ \left| \frac{\lambda^{(2q)}(\zeta|\ \beta)}{(2q)!\ k_q^2} \right| ,\ \zeta\ \in\ (a,b) \\ \leq \sup_{\eta' \in (a,b)} \left| \frac{\lambda^{(2q)}(\eta'|\beta)}{(2q)!\ k_q^2} \right| , \\ = GQB\ . \end{aligned} \tag{20}$$

Thus the largest value of GQB for physically realistic conditional intensities bounds the absolute difference between J_q and J, and hence between the approximated log-likelihood $\hat{\ell}_q$ and the actual log-likelihood ℓ.

In the following a model of the conditional intensity describing a neural refractory event is specified in order to (i) specify the $2q^{th}$ derivative of a realistic model of arguably the most dramatic physiological neural event that exists, and (ii) establish that the bound, GQB, for this model is small for a modest quadrature order q Fig. 3.

The argument is as follows. Conjecture the existence of a moment in time (the action potential threshold) when the physical system governing the dynamics of a neuron has entered into an effectively irreversible process generating an action potential. This threshold for action potential initiation depends on the intrinsic state of the neuron which may vary prior to the initiation of each spike (Henze and Buzsaki 2001). Thus, the probability of neural firing immediately after the reported time of a neural action potential is not zero and is better modeled as a continuous function rather than as a discontinuity. This consideration motivates the following sigmoidal function of the conditional intensity following an action potential. It is specified to capture an exceedingly rapid but continuous transition of the conditional intensity into the refractory epoch. Specifically, model the

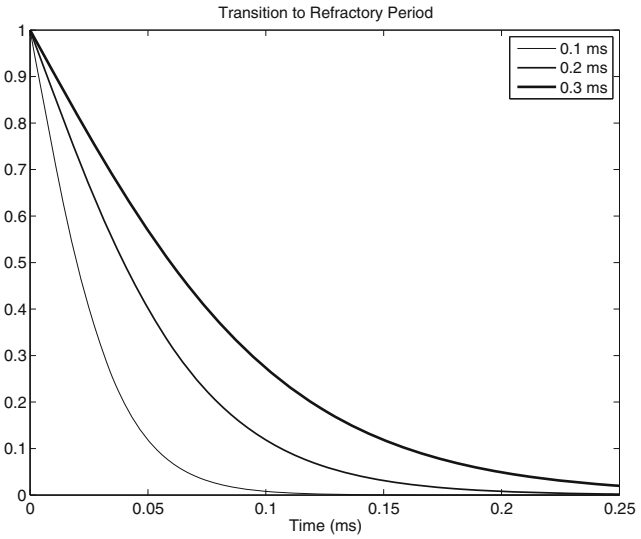

Fig. 2 The refractory model specified in Eq. 21 depicted for three choices of α. Here λ_0 is set equal to 1

conditional intensity λ_r, t seconds following an action potential as,

$$\lambda_r(t) = \frac{2\lambda_o}{1 + e^{\alpha t}}\ . \tag{21}$$

Then the conditional intensity decays from a maximal value of λ_o immediately following an action potential. Figure 2 depicts λ_r for suitable α.[5]

With the specification (21), the quadrature error bound GQB can be computed analytically. Let

$$g_q(t) = \left(1 + e^{\alpha t}\right)^{-q}\ . \tag{22}$$

In Appendix A, the j^{th} derivative, $D_j(t)$ of $\lambda_r(t|\beta)$ is shown to be:

$$D_j(t)\ =\ 2\lambda_o \alpha^j e^{\alpha t} \sum_{k=2}^{j+1} \alpha_{j,k}\ e^{(k-2)\alpha t}\ g_k(t)\ ,\ \ j\ >\ 0\ , \tag{23}$$

where

$$\alpha_{j,k} = \begin{cases} (k-1)\alpha_{j-1,k} - (k-1)\alpha_{j-1,k-1}\ , & k > 2, \\ & k - j < 1 \\ -(k-1)\alpha_{j-1,k-1} & , \ k > 2, \\ & k - j = 1 \\ -1 & , \ k = 2 \\ 0 & , \ k - j > 1 \end{cases}\ . \tag{24}$$

[5]The parameter α is specified such that the time for the conditional intensity to be near zero following an action potential is .1 ms, .2 ms, and .3 ms.

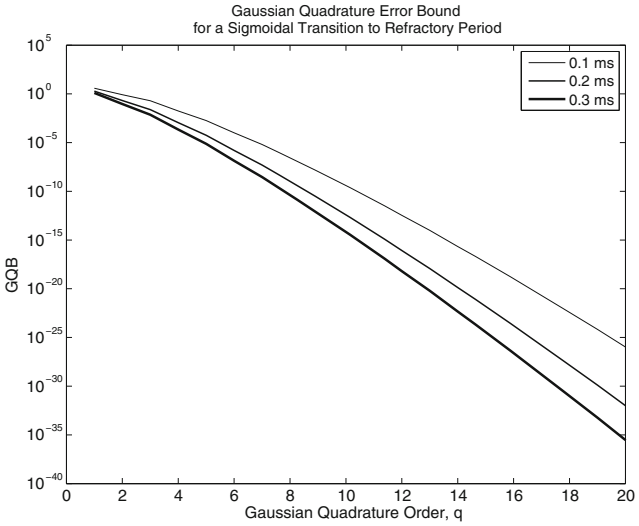

Fig. 3 Quadrature error due to a single refractory event becomes negligible with increasing order. Each curve is associated with the corresponding gray-scale curve depicted in Fig. 3

Then the Gaussian quadrature error is less than or equal to

$$GQB = \sup_{\eta' \in (a,b)} \left| \frac{D_{2q}(\eta')}{(2q)!\, k_q^2} \right| . \tag{25}$$

Figure 3 depicts GQB as a function of quadrature order.

For a refractory event in isolation, the analytic error bound (25) suggests that a quadrature order of 20 will yield ℓ_q within 10^{-20} of ℓ, the desired log-likelihood. Since Gaussian quadrature integrates order $2q-1$ polynomials exactly, a 39^{th} order polynomial accurately models the refractory effect. When λ_r is used as a factor in a multiplicative model of the conditional intensity, see Eq. (3), and used in Section 8, the full model is a polynomial of order ≥ 39.[6] This full model is an example member of the class of models discussed in Section 5.1. Thus, there is no extra difficulty using the proposed algorithm (Section 6) with accurate models of neural refractoriness.

6 Adaptive strategy for selecting quadrature order

The proposed maximum-likelihood estimate is computed iteratively. The steps are as follows:

1. Pick an initial quadrature order q.
2. Compute $\hat{\beta}_q$ and the corresponding conditional intensity estimate $\hat{\lambda}_q(t|\hat{\beta}_q)$, for a single trial.
3. Compute $\boldsymbol{\sigma}_q$, the vector of approximate standard errors. This vector is equal to the element-by-element square root of the diagonal of
$$\mathbf{C}_q(\hat{\beta}_q) = -\left[\mathbf{I}_q\left(\hat{\beta}_q\right)\right]^{-1} . \tag{26}$$
4. Using Eqs. (58) and (59) specified in Appendix C, compute the polynomial coefficients, g_j, $j = 0, \ldots, x$.[7]
5. Compute the largest absolute local extremum of the approximation, $\hat{\lambda}^{(2q)}$, by computing the roots, t_k, $k = 1, \ldots, x-1$ of the order $x-1$ polynomial,
$$\hat{\lambda}^{(2q+1)}(t|\hat{\beta}_q) \approx \sum_{j=0}^{x-1} (j+1)\, g_{j+1}\, t^j . \tag{27}$$
Discard complex-valued roots, and roots lying outside the interval $(-1, 1)$ to obtain $t_{k'}$, $k' = 1, \ldots, x'$.
6. By evaluating Eq. (59) at $t_{k'}$, $k' = 1, \ldots, x'$, compute the approximate local extremum, $e_{k'}$, scaled by $(2q)!$, and k_q^2 :[8]
$$e_{k'} \approx \left|\hat{\lambda}^{(2q)}(t_{k'}|\hat{\beta}_q)\right| \Big/ (2q)!\, k_q^2 . \tag{28}$$
7. Compute,
$$e_s = \max_k\, 2\sqrt{\frac{e_k}{(\boldsymbol{\sigma}_q)_k}} . \tag{29}$$
Based upon Eq. 88, e_s approximates the worst-case absolute error $\left|\beta - \hat{\beta}_q\right|$ with respect to parameter standard deviation.[9]
8. Compare e_s to a stopping threshold. For example, a stopping threshold of .1 indicates that the worst-case quadrature error across parameters is one-tenth of the standard error associated with the worst-case parameter.
9. Augment the quadrature order, q and repeat steps 2-8.

To assess the accuracy of e_s, it is useful to compute a related, quantity e_q, that does not involve scaling by the standard errors, $(\boldsymbol{\sigma})_{k'}$, $k' = 1, \ldots, x'$:

$$e_q = \max_k\, 2\sqrt{e_k} . \tag{30}$$

The quantity, e_q, is equal to GQB Eq. (20), in the case that $\hat{\lambda}_q$ is equal to the actual integrand and the extremum does

[6] The multiplication of a polynomial of order x with a polynomial of order y results in a polynomial of order $x+y$.

[7] The results in this paper are computed using x equal to 7. When x is increased there are three effects. The first is that deviations of $\hat{\lambda}_q^{(2q)}$ from zero are more accurately approximated by the expansion, Eq. (48). Second, the computation time of the expansion increases (though slightly when using a single trial). And third, Eq. (59) becomes more numerically unstable involving, in a sum, more terms over larger scales.

[8] This scaling is included to reduce rounding errors incurred when adding quantities of greatly differing scales.

[9] As mentioned in the introduction, in the situation where a minimum physiologically meaningful scale is known, $(\boldsymbol{\sigma}_q)_k$ can be replaced by this value.

Table 2 For $-\int_{-1}^{1} y^2/a^2 + y^2 dy$: Absolute quadrature error $|I - I_q|$, and absolute quadrature error approximate upper bound, e_q

a	$q = 10$		$q = 30$		$q = 50$		$q = 70$		$q = 150$	
	$\log_{10}(\lvert I\text{-}I_q\rvert)$	$\log_{10}(e_q)$	$\log_{10}(\lvert I\text{-}I_q\rvert)$	$\log_{10}(e_q)$	$\log_{10}(\lvert I\text{-}I_q\rvert)$	$\log_{10}(e_q)$	$\log_{10}(\lvert I\text{-}I_q\rvert)$	$\log_{10}(e_q)$	$\log_{10}(\lvert I\text{-}I_q\rvert)$	$\log_{10}(e_q)$
0.025	−1.2	0.4	−1.6	0.5	−1.9	0.3	−2.3	−0.02	−4.1	−1.4
.05	−1.1	0.7	−1.8	0.4	−2.7	−0.3	−3.6	−1.0	−7.0	−4.1
0.25	−2.1	−0.1	−6.4	−3.9	−10.7	−5.7	−14.7	−7.1	−14.7	−6.4
0.5	−3.9	−1.9	−12.3	−7.0	−16.0	−7.8	−15.1	−6.2	−14.8	−6.3
2.5	−13.8	−8.1	−15.8	−7.4	−16.3	−8.8	−15.7	−6.0	−15.4	−8.3

For this integral, the upper-bound e_q is always greater than the actual absolute quadrature error. As e_q tends to zero, $e_q/|I - I_q|$ increases. This trend is exhibited in Fig. 10 when applied to actual data. Together, the results suggest that the use of e_q in the computation of the stopping condition, Step 8, Section 6, does not result in iterations stopping early, but rather tends to increase the quadrature order q beyond that which is required

not lie on the boundary. In Table 2 the tightness of e_q is compared against the actual quadrature error for the integral,

$$I = -\int_{-1}^{1} \frac{y^2}{a^2 + y^2}\, dy\,, \tag{31}$$

$$= a\left[\tan^{-1}\left(\frac{1}{a}\right) - \tan^{-1}\left(\frac{-1}{a}\right)\right] - 2\,, \tag{32}$$

for various a and quadrature orders, q. Here $\hat{\lambda}_q$ is replaced by the actual integrand, $-y^2/a^2 + y^2$ to test the accuracy of e_q for a difficult integral. As a approaches zero the first derivative of the integrand tends to $-\infty$. For this integral, and for various a, and quadrature orders q, the Gaussian quadrature approximate upper bound e_q is always greater than the actual quadrature error, but is not tight, in that it over-estimates the actual quadrature error, see Table 2.

Steps 3-8 for a single iteration of the proposed algorithm is a computationally fast operation, owing to the use of only a single trial of the estimated conditional intensity, and because computation of the local extrema of the roots of an x-order polynomial ($x < 10$) is also fast. Note that in the situation where the initial q is chosen sufficiently large, this adaptive algorithm stops after a single iteration.

In the computation of e_q associated with Table 2 and the integral Eq. (32), e_q is computed using the actual integrand. In Step 6, e_s is computed from the estimate of the conditional intensity, $\hat{\lambda}_q$. This estimate adds extra uncertainty to the approximate upper-bound on the quadrature error. In Section 8, the accuracy of e_s is explored with actual neural data where the actual integrand (in this case, the conditional intensity) is not known.

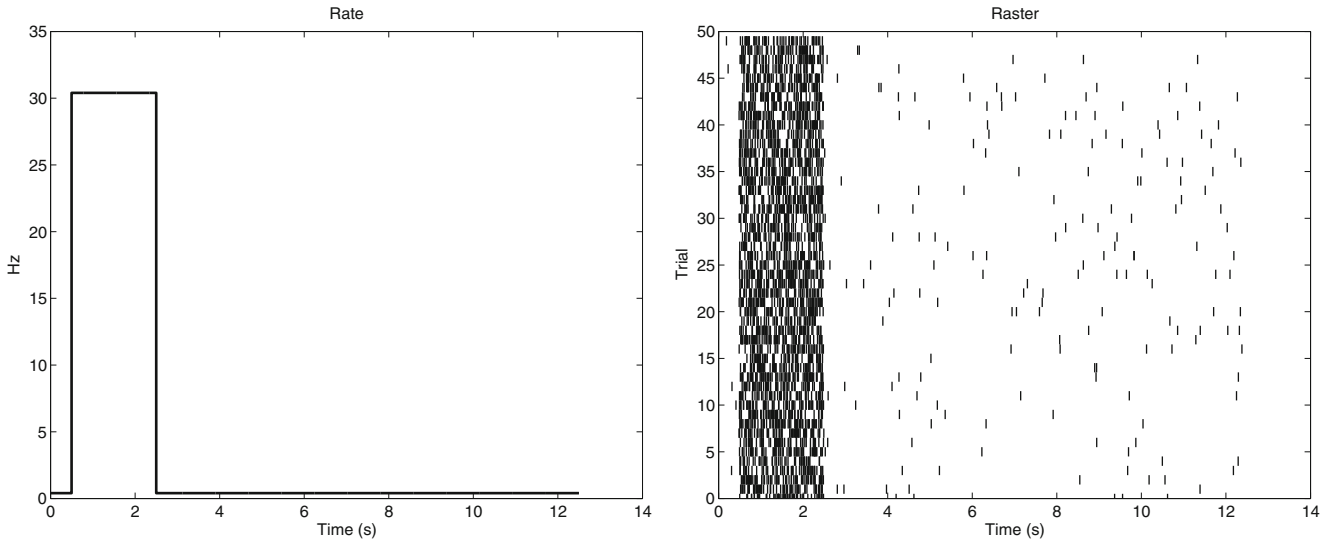

Fig. 4 Left: Simulated neuron activation times are drawn from an inhomogeneous Poisson process with the specified rate. Right: Example realization of the inhomogeneous Poisson process (rate specified on left). These simulated data are used to compare the proposed estimates using a continuous-time point process model to those computed using a discrete-time model

7 Simulation

To demonstrate the efficacy of the proposed fast continuous-time MLE, a time-cell with a rate of spiking depicted in Fig. 4 is simulated to obtain neuron activation times for fifty trials undergoing Poisson spiking. This neuron is observed for four durations of observation, ranging from 2.5 seconds to 25 seconds. For each duration of observation, recordings for fifty trials are taken, and from the recorded spike times the proposed approximate continuous time MLE is computed, $\hat{\beta}_q$ with $q = 20, 40$, as well as the more standard MLE computed using a discrete-time point process model. These estimates are computed for polynomial models of the neural spiking rate, with orders ranging from 3 to 10. Representative fits are shown in Fig. 5. Table 3 gives the time to compute for the various simulation scenarios.

The proposed continuous-time MLEs are computed much more quickly than their more standard discrete-time counter-parts, with, in this example, a maximal computation time advantage achieved for an observation of 25 seconds modeled with a 10^{th} order polynomial. In this case 1.6 seconds are required to compute the continuous-time estimate while the discrete-time version requires 10.1 seconds to form the model-matrix and 24.1 seconds to compute the discrete-time estimate for a total computation time of 34 seconds.[10] The dependence upon the number of model parameters (p), the number of discretized bins (n, discrete-time model) and the Gaussian quadrature order (q, continuous-time model) is further demonstrated in Figs. 6 and 7.

8 Hippocampal time-cells

The fast continous-time point process estimation introduced in this work is applied to tetrode recordings from single neuron activity in the rodent hippocampus (MacDonald et al. 2011). In this experiment, rats were trained to distinguish sequences of two events separated in time by a delay. For each trial, one of two objects is first presented. Following an empty ten second gap (a delay), one of two odours is presented (see Fig. 1 in Kesner et al. (2005)). Electrophysiological data was collected over many trials and on each trial the rat had to remember the first event in the event sequence in order to respond appropriately to the second event in the event sequence. Upon appropriately performing the trial, the rat received a reward. This experimental paradigm provides an opportunity to explore how neurons encode a sequence of

[10] Using the MathWorks MATLAB glmfit() command to compute the discrete-time parameter estimates. This function uses a version of iterated re-weighted least squares. All computations in this work are performed using a laptop equipped with an Intel P8600 Core2-Duo processor running at 2.4 GHz.

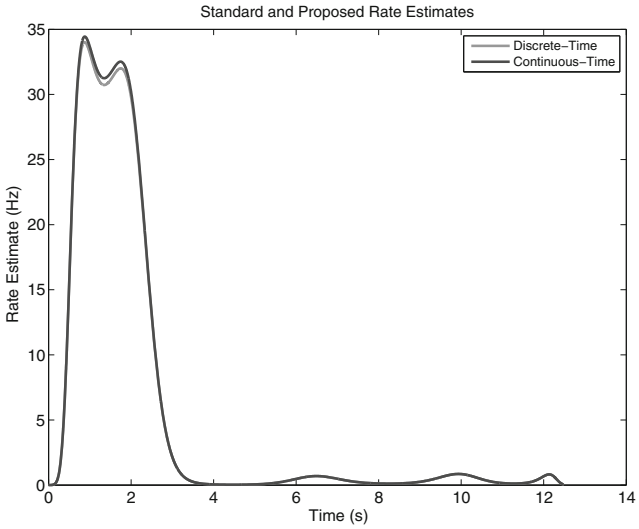

Fig. 5 Proposed (continuous-time) and more standard point-process rate estimates (discrete-time). Despite the modeling difference between continuous and discrete-time point process models, the rate estimates are similar. Though the estimates are similar the computation times vary markedly (Table 3)

events and how their activity bridges an identical temporal gap shared by distinct event sequences. One of the striking findings of this study is that individual neurons become active at different times during the delay between events and that they activate sequentially to bridge this delay (MacDonald et al. 2011). In this study the long delay sequence required the use of $n \approx 10^4$ time-bins to perform inference with a discrete-time point process model of neural activity.

To probe the effectiveness of the proposed methodology MLE estimates are computed from this data using both continuous-time and discrete-time point process models of neural activity. In these models of the conditional intensity, a tenth-order polynomial is used to account for temporal changes in the probability of firing, as well as history dependence in the form of a refractory period. When using the continous-time point process model, the multiplicative component, λ_r, of the conditional intensity model, λ immediately following a neural spiking event is modeled according to Eq. (2), and is specified by the history kernel γ. Here γ is specified to be refractory:

$$\gamma(t) = \beta\left[s(t)r(t) + (1 - s(t))\, e(t)\right], \tag{33}$$

where

$$s(t) = \left[1 + e^{\alpha_s (t - s_0)}\right]^{-1}, \tag{34}$$

$$r(t) = \left[1 + e^{\alpha_r (-t + r_0)}\right]^{-1}, \tag{35}$$

$$e(t) = \left[1 + e^{\alpha_e (t - e_0)}\right]^{-1}, \tag{36}$$

Table 3 Simulated data: computation times

Poly. Model Order	(in Seconds)															
	2.5 Second Observation				6.0 Second Observation				12.5 Second Observation				25.0 Second Observation			
$(p-1)$	$\hat{\beta}_q^{(20)}$	$\hat{\beta}_q^{(40)}$	$\hat{\beta}_r$	Setup	$\hat{\beta}_q^{(20)}$	$\hat{\beta}_q^{(40)}$	$\hat{\beta}_r$	Setup	$\hat{\beta}_q^{(20)}$	$\hat{\beta}_q^{(40)}$	$\hat{\beta}_r$	Setup	$\hat{\beta}_q^{(20)}$	$\hat{\beta}_q^{(40)}$	$\hat{\beta}_r$	Setup
3	0.6	0.6	1.4	0.8	0.6	0.6	1.6	1.9	0.9	0.8	3.3	4.3	0.7	0.7	6.9	8.7
7	1.0	1.1	1.4	0.8	1.2	1.2	3.7	2.2	1.2	1.3	7.4	4.8	1.0	1.1	15.0	9.7
10	1.2	1.3	2.4	0.9	1.5	1.6	5.8	2.2	1.7	1.8	12.5	5.0	1.5	1.6	24.1	10.1

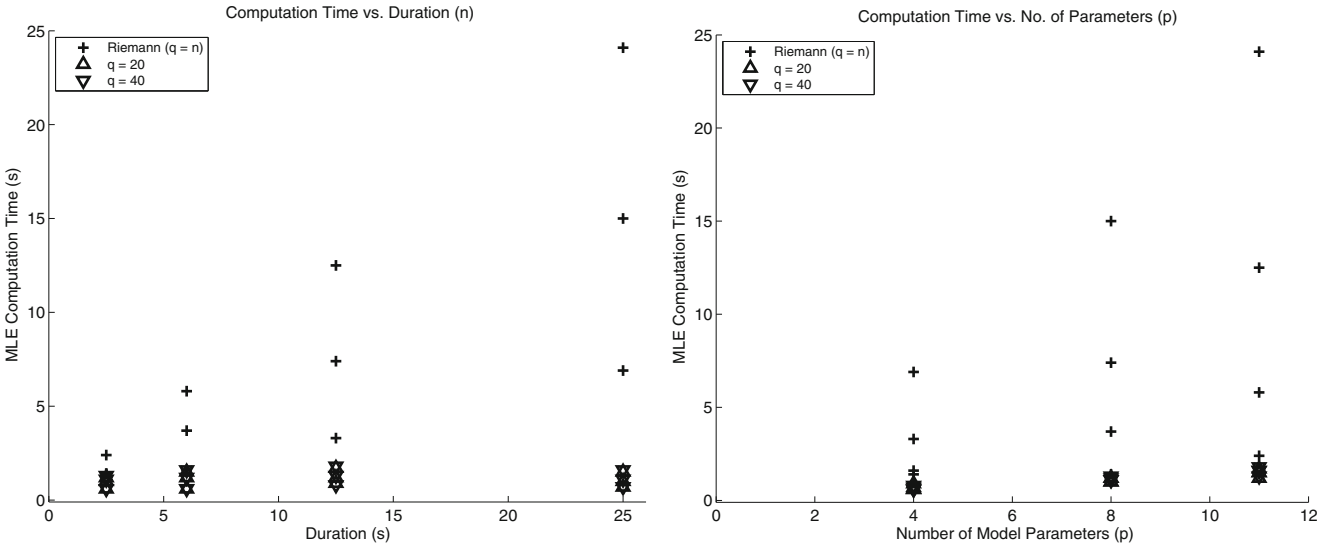

Fig. 6 The proposed methodology is much faster. The time to compute the MLE of the discrete-time model parameters (+) depends strongly on both n and p and is up to an order of magnitude larger than the time to compute the continous-time model parameter MLEs (*triangles*). The computation times are listed in Table 3

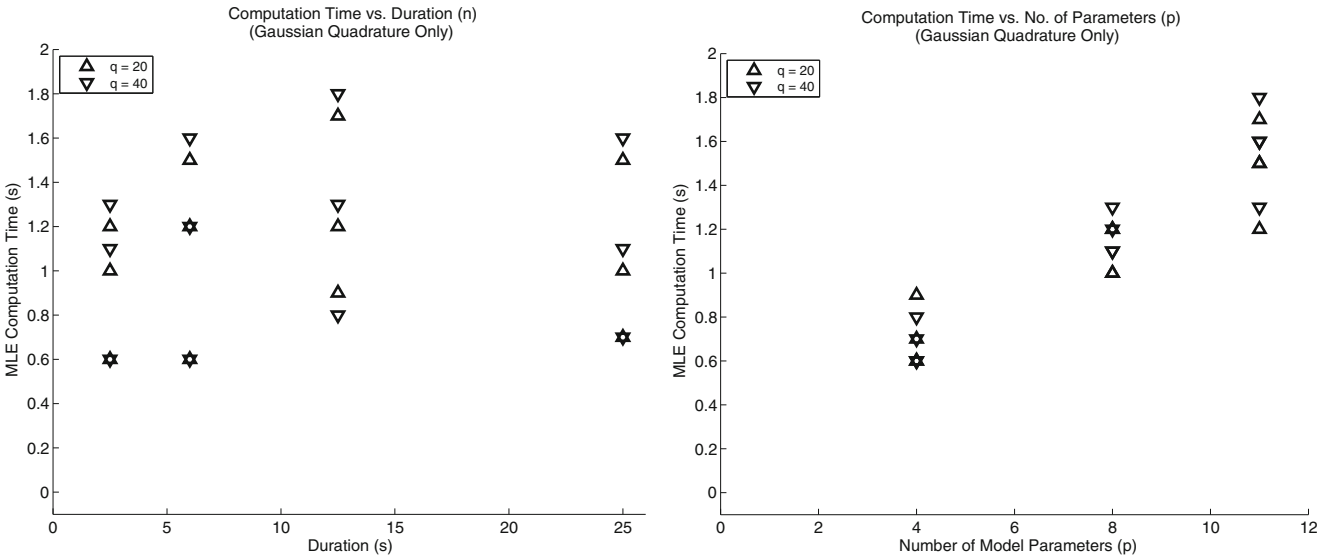

Fig. 7 The computation time for the continuous-time model parameters depends weakly on duration. Plotted quantities are as in Fig. 6 but excluding the computation times for the discrete-time model parameters

with s_0 equal to 1 ms, r_0 equal to 0.05 ms, and e_0 equal to 1.8 ms. Thus specified, γ is a logistic function approximation to the box function that is one on the interval $(0, 2)$ ms and zero otherwise. The parameters are chosen consistent with Fig. 2, where the plot of a realistic entry into a refractory epoch is plotted (see Section 5.3). Here the parameter β controls the size of the refractory effect and is estimated from data. Because γ does not depend linearly upon the alphas, they are not parameters in a model belonging to the class of models considered in this work (see Eq. (2)). The discrete-time counter-part, $\lambda_{r,t}$, is

$$\log \lambda_{r,t}(\beta) = \beta_1 y_{t-1} + \beta_2 y_{t-2} \,. \tag{37}$$

Here β_1 and β_2 are parameters weighting the lagged counts y_{t-1}, and y_{t-2}. Using a 1 ms discretization, a 2 ms refractory period is modeled.

Raster plots of example hippocampal cell activity are provided in Fig. 8 (top). The temporally localized nature of the firing activity for these cells is evident. The associated conditional intensity estimates are depicted (Fig. 8, bottom) for the first trial. The difference between the estimates computed with the fast continous-time model, and the slower discrete-time model is difficult to discern on the scale plotted.

The effect of discretization on estimates of the history effect is shown in Fig. 9. The continuous-time model avoids

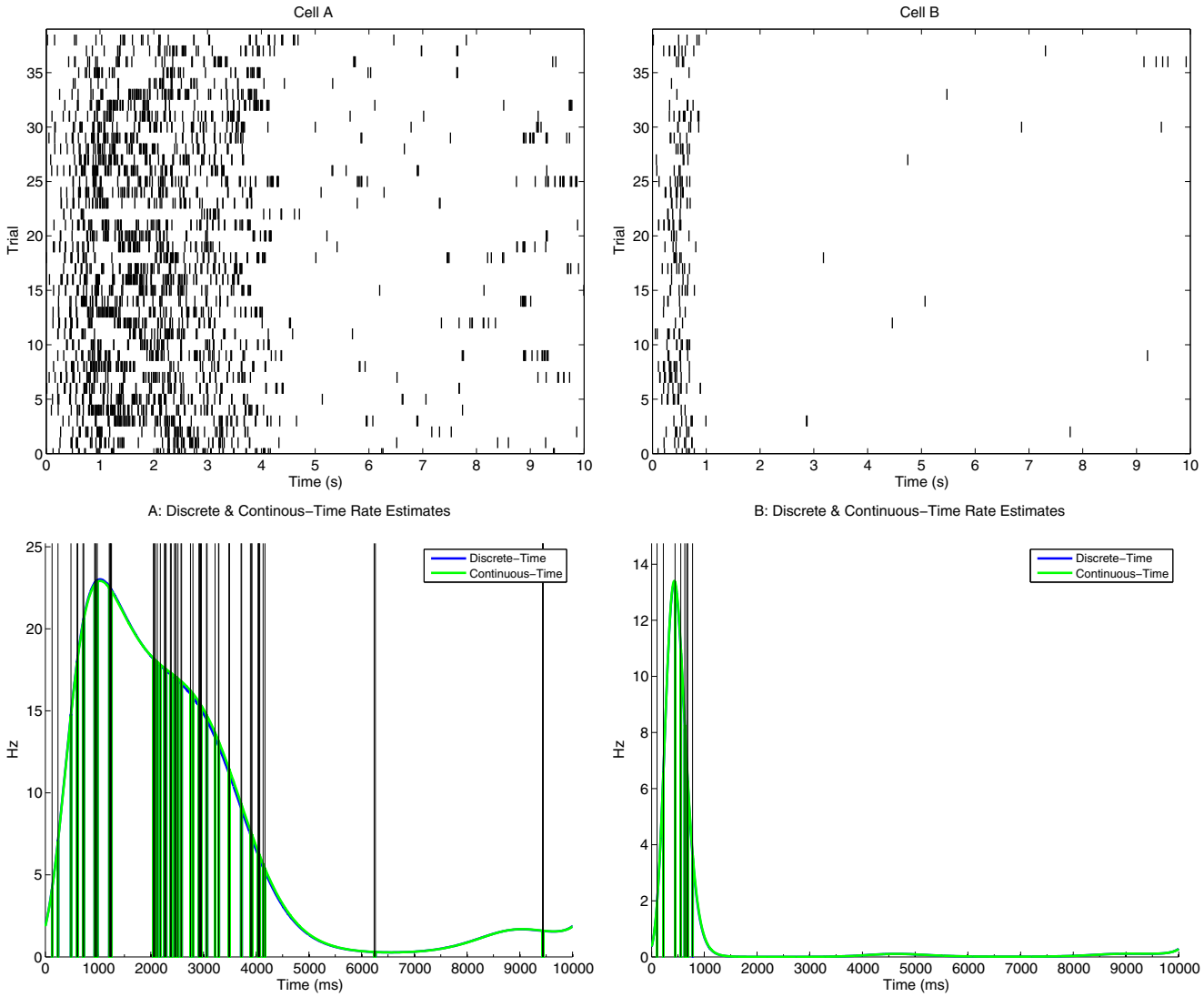

Fig. 8 Top: Time-cells exhibit temporally localized firing activity. Black vertical streaks demark putative single-cell action potentials. For these example cells, firing activity occurs preferentially at the beginning of the delay epoch. Bottom: Associated conditional intensity estimates for the first trial. Black vertical lines indicate cell spike times. The difference between the discrete-time and continuous-time estimates is difficult to see

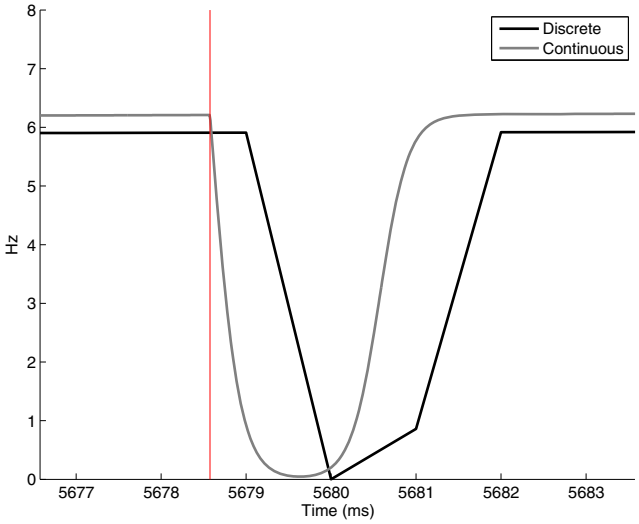

Fig. 9 Increased accuracy via continuous-time modeling of the refractory effect. The continuous-time model allows for a more accurate characterization of the time immediately following a spiking event. Here the discrete-time model lags the actual spike time due to the use of 1 *ms* binning. In this case, the difference in history effect manifests in a slight offset of the conditional intensity, before and after the neural spiking event. Note that for a finer time discretization the difference between the models will be reduced. This improvement incurs an increased computational cost associated with the discrete-time model

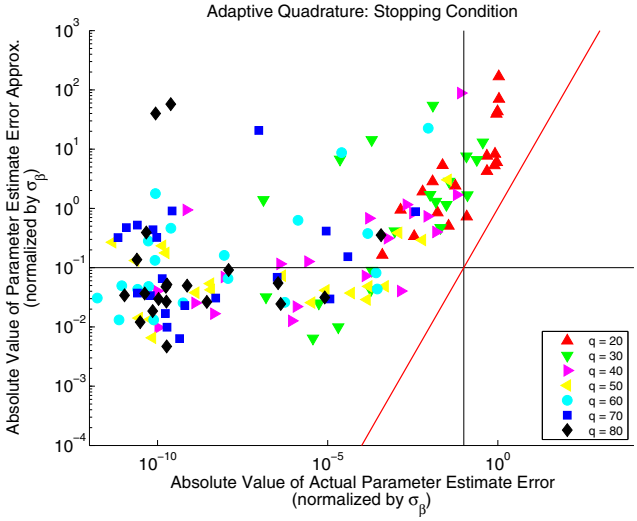

Fig. 10 The adaptive stopping condition e_s is conservative. Each mark corresponds to a single cell and quadrature order, and plots e_s against a proxy for the quadrature-error-free stopping condition. See text for details. Marks on the red line correspond to the situation where there is no quadrature-error (see text – assuming a $q = 100$ results in no quadrature error). For all hippocampal cells, e_s is greater than the actual stopping condition associated with no quadrature error. For a stopping threshold equal to 0.1, the black lines (horizontal and vertical) demark decision boundaries separating correct decisions from mistakes. Marks in the upper right quadrant correspond to not stopping adaptive iteration when the iterations should not be stopped. Marks in the upper-left quadrant correspond to not stopping when the iterations should be stopped. Marks in the lower-left quadrant indicate correctly stopping adaptive iteration, and marks in the lower-right quadrant indicate stopping adaptive iteration when it should not be stopped. This latter condition is the most serious. In this situation, parameter estimate error due to quadrature will contribute to more than one-tenth of a standard deviation to the value of the parameter estimate. Note that this does not occur. This, together with marks in the upper-left quadrant indicate that the adaptive procedure is conservative. Also see Tables 2 and 4

a latency exhibited by the MLE computed using a discrete-time point process model.

In Fig. 10 the utility of the approximate upper-bound on parameter estimate error e_s, specified in Eq. (29), is compared to a proxy. This proxy is a worse-case quadrature error approximation computed assuming parameter estimates computed using a quadrature order of 100 are exact. Specifically, the proxy (the horizontal axis in Fig. 10) is the maximum absolute scaled difference between parameters estimated using a quadrature order equal to 100 with the parameter estimates computed using a smaller quadrature order. The per-parameter scaling is the per-parameter standard error. Thus the proxy represents the worse-case actual quadrature error with respect to the associated estimator standard deviation. It is the adaptive quadrature stopping condition (see Section 6), that results in the situation where there is little quadrature error.[11] Similarly, the vertical axis in Fig. 10 is e_s. Thus, the iterations in the adaptive strategy (Section 6) are stopped whenever a mark in Fig. 10 is less than a user defined stopping threshold (for e.g., a value of 0.1). Each mark in Fig. 10 corresponds to a single hippocampal-cell analyzed with a specific quadrature order. Figure 10 suggests that the adaptive stopping condition based upon e_s (Section 6) is conservative. See the caption for details.

[11] No quadrature error in the case where q equal to 100 is sufficient to exactly integrate the conditional intensity.

In Table 4, the quadrature order yielding a stopping condition of .1 is provided for each of the analyzed hippocampal cells. The adaptive procedure results in relatively low-order quadrature orders for many cells.

In Fig. 11 the computation time to estimate the MLEs for the continuous-time model and discrete-time model are plotted. For these recordings, the continuous-time MLE is computed with a Gaussian quadrature order of 60, as well as the approximate quadrature error bound e_q, Eq. (30). Here the quadrature order is chosen roughly from the modeling considerations presented in Section 5. Note that for q equal to 60, e_s, specified in Eq. (29), is less than 0.1 for all but one cell (see Table 4). Computation typically requires 4-7 times less time to compute the MLE with the continuous-time model than with the discrete-time counter-part. Computation times are given for the MLE of 19 rodent hippocampal cells recorded and analyzed in MacDonald et al. (2011). The full experiment involves more than 80 of these cells distributed across four rodents. The full computation

Table 4 The adaptive algorithm, Section 6, yields low order quadratures for many of the hippocampal cells

Cell	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Stopping Order (q)	50	30	30	30	30	40	40	40	60	40	40	50	50	50	60	30	50	30	50	100

Here, the initial quadrature order is set to 10, and the quadrature order, q is incremented by 10 after each iteration. See Fig. 10 and Table 2 for the performance of the iterative stopping condition

time for a single discrete-time model is around one hour, and with the continuous-time model it is around ten minutes.

The time-rescaled inter-spike intervals (ISIs) can be used to assess model fit. Following the description of the QQ-plot detailed in Brown et al. (2002), Truccolo et al. (2005), in Fig. 12, the quantiles of the time-rescaled and transformed ISIs computed using the discrete-time model (solid with plus marks), and computed using the continuous-time model (solid) are plotted against the theoretical quantiles of the rescaled and transformed ISIs under the hypothesis that the model is correctly specified. There is no appreciable difference between the QQ-plots for the two models (see Fig. 12). In this situation, goodness-of-fit assessment based upon the QQ-plot is identical when using either the discrete-time model or when using the continous-time model. Here the ISI rescaling is performed using Riemann quadrature for the discrete-time model, and is accomplished using Gaussian quadrature (with a q equal to 30), when using the continuous-time point process model.

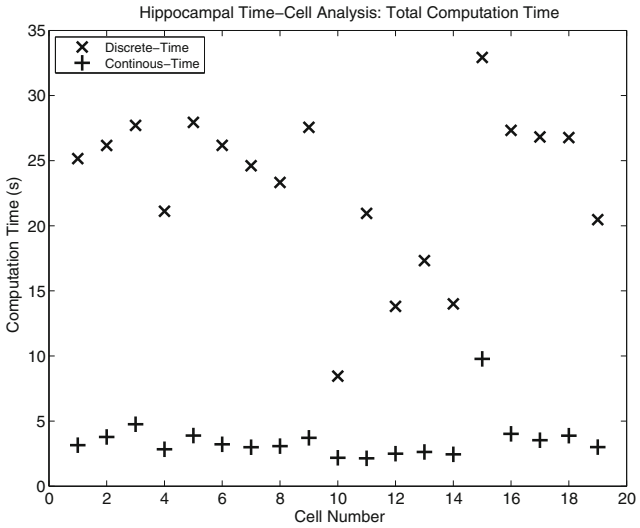

Fig. 11 The continuous-time MLE (quadrature order 60) typically requires 4-7 times less time to compute than the discrete-time counterpart. Computation times are given for the MLE of 19 rodent hippocampal cells recorded and analyzed in MacDonald et al. (2011). The full experiment involves more than 80 of these cells distributed across four rodents. The full computation time for a single discrete-time model is around one hour, and with the continuous-time model it is around ten minutes. Both the discrete-time and continuous-time conditional intensities are modeled as tenth-order polynomials, with history effects specified by Eq. (37) (discrete-time model) and Eq. (33) (continuous-time model). The time to compute includes the time to form the model matrices. Note that cell number 15 exhibits a continuous 23 Hz firing rate and is likley multi-unit. In this case $|T_s|$ for an average trial is 231 and is about 4 times larger than q. There are 39 trials

9 Discussion

In Section 5 the accuracy of the proposed approximate MLE is addressed. By studying a plausible model of the neural refractory effect (Section 5.3), the existence of a plausible conditional intensity model that can be accurately integrated using Gaussian quadrature of modest order is established.

In Appendix C, a cheaply-computed bound on parameter estimate inaccuracy due to quadrature error is provided. Thus effects due to quadrature error can be monitored, and the quadrature order can be appropriately adjusted. Because the quadrature is computationally cheap, large changes in quadrature order can be employed.

In simulation (Section 7), and when analyzing actual data (Section 8), the time to compute the maximum likelihood estimates using Gaussian quadrature and the continuous-time point process models of neural activity (Section 2),

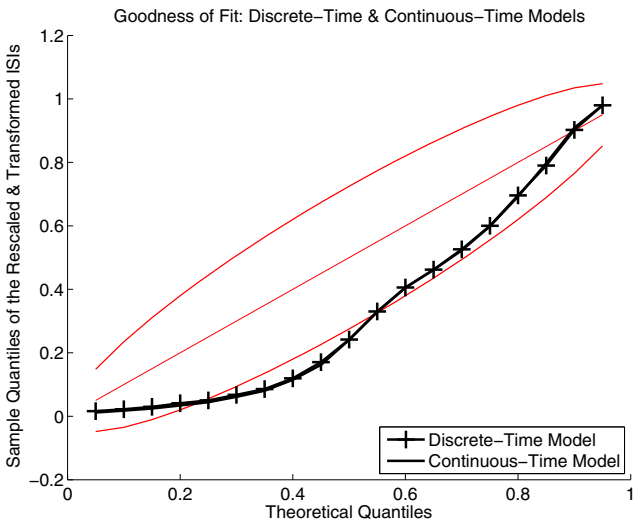

Fig. 12 Example QQ-plots constructed using the discrete-time (plus symbol) and continuous-time (solid) point process models. The curves are computed using the data recorded from cell 5 in Fig. 11. The two curves are nearly identical on the scale plotted. The thick red lines indicate approximate 95 percent confidence bounds

is much faster than their discrete-time counter-parts. These results are conservative, in the sense that for all calculations, the Gaussian quadrature nodes and weights are re-computed. This time can be reduced, by storing these values offline, or by employing the methods presented in (Hale and Townsend 2013). These approaches are left for future study. They will be most important for the small p and large q case.

Three limitations to this work are (i) the number of computations required depends upon the total number of spikes due to the first term in Eq. (2) (see Fig. 11), (ii) for non-identifiable log-likelihood functions, the standard issues arise, and (iii) when considering oscillatory conditional intensities, such as those investigated in Lepage et al. (2013), limited speed-up may occur due to the fact that the integral is zero for an integer number of observed oscillatory periods. In case (ii), there exist parameter values that yield the same log-likleihood value (non-identifiable case), or similar values (approximately non-identifiable case, and often a numerically unstable case). These issues have been well documented in the statistics literature. They are discussed in books on generalized linear modeling. See for e.g., (McCullagh and Nelder 1999). In case (iii), a simple two-step procedure is available: estimate the parameters assuming the integral in Eq. (2) is zero. Then with this parameter estimate as a starting guess, compute a few iterations including the integral.

In this work focus is placed upon significantly reducing n to q in the quadrature approximation to the integral of the conditional intensity appearing in the log-likelihood, Eq. (2), while maintaining sufficient quadrature accuracy. Emphasis is placed upon determining q such that quadrature error is acceptable.

In Section 6, a quadrature order selection criterion based upon a previously unreported approximation to the higher order derivative of the integrand (Appendix C) is introduced. Its use is motivated mathematically (Appendix C), its performance is demonstrated on a difficult integral (Table 2), and its utility is demonstrated when analyzing actual neural recordings (Fig. 10 and Table 4). It is expected that when used to analyze future neural recordings, this algorithm will be usefully employed to reduce the risk of unacceptable error due to Gaussian quadrature.

The statistical analysis of neural spiking is central to neuroscience. As experiments increase in size and complexity the speed at which these analyses can be conducted becomes increasingly important. In this work, by replacing discrete-time conditional intensity models with their continuous-time counter-parts and employing the incredibly efficient Gaussian quadrature, the computational complexity is reduced from $O(np^2)$ to $O(qp^2)$, memory requirements are reduced from $O(np)$ to $O(qp)$, and any discretization-error is eliminated. This is accomplished without reducing the available statistical methodology. The statistical methodology used to perform data analysis with discrete-time point process models of neural activity established in, for e.g., (Truccolo et al. 2005; Lepage et al. 2012) is applicable to the exponential family of continuous-time point process models used in this work.

A typical statistical analysis is an iterative exercise. By reducing the time to compute by even a factor of two, two weeks of model fitting becomes one. This contribution is expected to be a welcome addition to those performing statistical inference with point process models of neural activity.

Conflict of interests The authors declare that they have no conflict of interest.

Appendix A: higher-order derivatives of the refractory model

Assume $D_j(t)$ specified in Eq. (23) is valid. Then, for $j > 0$,

$$\begin{aligned} D'_j(t) &= \frac{d}{dt}\left\{2\lambda_0\alpha^j e^{\alpha t}\sum_{k=2}^{j+1}\alpha_{j,k}e^{(k-2)\alpha t}g_k(t)\right\}, \\ &= 2\lambda_0\alpha^{j+1}e^{\alpha t}\sum_{k=2}^{j+1}\alpha_{j,k}e^{(k-2)\alpha t}g_k(t) \\ &\quad +2\lambda_0\alpha^j e^{\alpha t}\sum_{k=2}^{j+1}\alpha_{j,k}(k-2)\alpha e^{(k-2)\alpha t}g_k(t) \\ &\quad +\alpha_{j,k}e^{(k-2)\alpha t}(-k)\alpha e^{\alpha t}g_{k+1}(t), \end{aligned} \tag{38}$$

since $g'_k(t) = -k\alpha e^{\alpha t}g_{k+1}(t)$. Collecting terms,

$$\begin{aligned} D'_j(t) &= 2\lambda_0\alpha^{j+1}e^{\alpha t}\sum_{k=2}^{j+1}(k-1)\alpha_{j,k}e^{(k-2)\alpha t}g_k(t) \\ &\quad -k\alpha_{j,k}e^{(k-1)\alpha t}g_{k+1}(t), \\ &= 2\lambda_0\alpha^{j+1}e^{\alpha t}\left[\sum_{k=2}^{j+1}(k-1)\alpha_{j,k}e^{(k-2)\alpha t}g_k(t)\right. \\ &\quad \left.-\sum_{k'=3}^{j+2}(k'-1)\alpha_{j,k'-1}e^{(k'-2)\alpha t}g_{k'}(t)\right], \\ &= 2\lambda_0\alpha^{j+1}e^{\alpha t}\sum_{k=2}^{j+2}\alpha_{j+1,k}e^{(k-2)\alpha t}g_k(t), \\ &= D_{j+1}(t), \qquad (j>0). \end{aligned} \tag{39}$$

The induction argument is completed by verifying Eq. (23) for $j = 1, 2, 3$ by direct calculation.

Appendix B: Non-orderly discrete-time "Gaussian quadrature" process

Gaussian quadrature might be used to compute the MLE associated with a discrete-time process. Based upon the Gaussian quadrature nodes t_j, $j = 1, \ldots, q$, consider the increment process, $Y_j = N(t_j) - N(t_{j-1})$. Unlike in the previous discussion of a discrete-time point process, here the duration, Δ_j, of the j^{th} increment is not constant, but rather equals $t_j - t_{j-1}$. This duration may be relatively large, and the orderliness property of the process is not guaranteed. For a given sample path of the process, let the number of observed counts in the j^{th} increment be y_j. The associated log-likelihood, ℓ_y, can be shown to equal,

$$\ell_y(\beta) = \sum_{j=1}^{q} y_j \log(\Delta_j \lambda(t_j|\beta)) - \sum_{j=1}^{q} \Delta_j \lambda(t_j|\beta) - \sum_{j=1}^{q} \log(y_j!) \,. \tag{40}$$

While Eq. (40) may approach the approximation Eq. (11) to the continuous-time log-likelihood Eq. (4) in certain situations, in general these approximations differ. To what extent and under what conditions equivalent inference can be conducted with either of the spike train models are questions appropriate for a future study.

Appendix C: Approximate $2q^{th}$ derivative of the conditional intensity

An approximation of $\hat{\lambda}_q^{(2q)}$ is provided in the following derivation. Begin by expanding $\hat{\lambda}_q(t|\hat{\beta}_q)$ in terms of the order $q' = 2q + x$ Legendre polynomials, where x is a user defined quantity specified in Section 6. The choice of x is discussed in Footnote 5 (Section 6). That is, compute the coefficients c_j, $j = 0, \ldots, q' - 1$, using order q' Gaussian quadrature:

$$\begin{aligned} c_j &= \sum_{j'=0}^{q'-1} w_{j'} \hat{\lambda}_q\left(t_{j'}|\hat{\beta}_q\right) L_j(t_{j'}) \Big/ \sum_{j'=0}^{q'-1} w_{j'} L_j^2(t_{j'}) \,, \\ &\approx \int_0^T \hat{\lambda}_q\left(t|\hat{\beta}_q\right) L_j(t)\, dt \Big/ \int_0^T L_j^2(t)\, dt \,. \end{aligned} \tag{41}$$

Here j indexes the Legendre polynomials, while j' indexes the roots, $t_{j'}$, of the q'^{th} order Legendre polynomial, $L_{q'}$. Then,

$$\hat{\lambda}_q(t|\hat{\beta}_q) \approx \sum_{j=0}^{q'} c_j L_j(t) \,. \tag{42}$$

For $j > 2q - 1$, $c_j L_j(t)$ is a polynomial that may not be exactly integrated by Gaussian quadrature of order q.

A direct computation of $\hat{\lambda}_q^{(2q)}$ from Eq. (42) is often inaccurate with standard double-precision floating point numbers. The sum in Eq. (42) often involves the sum of terms that span more than sixteen orders of magnitude for the case where q is equal to 40. When this occurs, numerical error results, and often leads to unacceptable inaccuracy. The problem is mitigated by deriving an alternate expression for $\hat{\lambda}_q^{(2q)}$. Proceed by Taylor expanding $\hat{\lambda}_q(t|\hat{\beta})_q$ about $t = 0$ to order $2q + x$:

$$\hat{\lambda}_q(t|\hat{\beta}_q) = \sum_{u'=0}^{2q+x} \frac{\hat{\lambda}_q^{(u')}(0|\hat{\beta}_q)}{u'!} t^{u'} \,. \tag{43}$$

From Eq. (43) the u'^{th} derivative is

$$\begin{aligned} \hat{\lambda}_q^{(u')}(t \mid \hat{\beta}_q)\,|_{t=0} &\approx \sum_{j'=u'}^{2q+x} \frac{j'!}{(j'-u')!} \frac{\hat{\lambda}_q^{(j')}(0|\hat{\beta}_q)}{j'!} t^{j'-u'} \,, \\ &= t^{-u'} \sum_{j'=u'}^{2q+x} \frac{j'!}{(j'-u')!} \frac{\hat{\lambda}_q^{(j')}(0|\hat{\beta}_q)}{j'!} t^{j'} \,, \end{aligned} \tag{44}$$

the fact that $\hat{\lambda}_q(t)$ is a polynomial is used to set the lower bound in the sum. From Eq. (42) the u'^{th} derivative evaluated at $t = 0$, is,

$$\hat{\lambda}_q^{(u')}(0|\hat{\beta}_q) \approx \sum_{j=u'}^{q'-1} c_j L_j^{(u')}(0) \,. \tag{45}$$

Here $L_j^{(u')}$ is the u'^{th} derivative of the order j Legendre polynomial. Because it can be shown, beginning with Rodriguez's formula, that $L_n^{(q)}$ is equal to,

$$\begin{aligned} L_n^{(q)}(t) = \frac{(n+q)!n!}{2^n} \sum_{k=q}^{n} \sum_{\ell=0}^{k-q} \sum_{\ell'=0}^{n-k} -1^{n-k-\ell'} \\ \times \frac{t^{\ell+\ell'}}{(n+q-k)!\,k!\,\ell!\,\ell'!\,(k-q-\ell)!\,(n-k-\ell')!} \,, \end{aligned} \tag{46}$$

$L_n^{(q)}(0)$ can ben determined analytically. At $t = 0$ in Eq. (46), only $\ell + \ell' = 0$ terms contribute:

$$\begin{aligned} L_n^{(q)}(0) &= \frac{(n+q)!n!}{2^n} \sum_{k=q}^{n} \sum_{\ell=0}^{k-q} \sum_{\ell'=0}^{n-k} \delta_{\ell,0}\delta_{\ell',0} - 1^{n-k-\ell'} \\ &\quad \times \frac{1}{(n+q-k)!k!\ell!\ell'!(k-q-\ell)!(n-k-\ell')!} \,, \\ &= \frac{(n+q)!n!}{2^n} \sum_{k=q}^{n} \frac{-1^{n-k}}{(n+q-k)!k!(k-q)!(n-k)!} \,. \end{aligned} \tag{47}$$

Equations (44), (45) and (47) combine to produce,

$$\begin{aligned}\hat{\lambda}_q^{(2q)}(t|\hat{\beta}_q) &= t^{-2q}\sum_{j'=2q}^{2q+x}\frac{j'!}{(j'-2q)!}\frac{\hat{\lambda}_q^{(j')}(0|\hat{\beta}_q)}{j'!}t^{j'}\ ,\\ &= t^{-2q}\sum_{u=0}^{x}\frac{1}{u!}\hat{\lambda}_q^{(u+2q)}(0|\hat{\beta}_q)t^{u+2q}\ ,\\ &= \sum_{u=0}^{x}\frac{t^u}{u!}\,\hat{\lambda}_q^{(u+2q)}(0|\hat{\beta}_q)\ , \end{aligned} \tag{48}$$

Continuing,

$$\begin{aligned}&\hat{\lambda}_q^{(2q)}(t|\hat{\beta}_q) =\\ &= \sum_{u=0}^{x}\frac{1}{u!}t^u\sum_{j''=u+2q}^{2q+x}c_{j''}L_{j''}^{(u+2q)}(0)\ ,\\ &= \sum_{u=0}^{x}\frac{1}{u!}t^u\sum_{j=u}^{x}c_{j+2q}L_{j+2q}^{(u+2q)}(0)\ ,\\ &= \sum_{u=0}^{x}\frac{1}{u!}t^u\sum_{j=u}^{x}c_{j+2q}\frac{(j+u+4q)!(j+2q)!}{2^{j+2q}}\\ &\quad\times\sum_{k=u+2q}^{j+2q}\frac{-1^{j+2q-k}}{(j+u+2q+2q-k)!k!(k-2q-u)!(j+2q-k)!}\\ &= \sum_{u=0}^{x}\frac{t^u}{u!}\sum_{j=u}^{x}c_{j+2q}\frac{(j+u+4q)!(j+2q)!}{2^{j+2q}}\\ &\quad\times\sum_{k'=0}^{j-u}\frac{-1^{j-u-k'}}{(j+2q-k')!\,(k'+u+2q)!\,(k')!\,(j-u-k')!} \end{aligned} \tag{49}$$

The terms in the last sum in Eq. (49) are symmetrical about the center index ($j-u$ even), or about the center indices ($j-u$ odd). For example, the first and the last terms are equal. When $j-u$ is odd, the signs alternate for terms that are identical in absolute-value and the sum is equal to zero. When $j-u$ is even, there are an odd number of terms in the sum. Consider the case, $j-u$ equal to four:

$$\begin{aligned}&\sum_{k'=0}^{4}\frac{-1^{j-u-k'}}{(j+2q-k')!\,(k'+u+2q)!\,(k')!\,(j-u-k')!}\\ &= \frac{1}{(j+2q)!\,(u+2q)!\,0!\,4!}\\ &\quad-\frac{1}{(j+2q-1)!\,(u+2q+1)!\,1!\,3!}\\ &\quad+\frac{1}{(j+2q-2)!\,(u+2q+2)!\,2!\,2!}\\ &\quad-\frac{1}{(j+2q-3)!\,(u+2q+3)!\,3!\,1!}\\ &\quad+\frac{1}{(j+2q-4)!\,(u+2q+4)!\,4!\,0!}\ . \end{aligned} \tag{50}$$

Because $j-u$ equals 4, the last term in Eq. (50) is,

$$\frac{1}{(u+2q)!\,(j+2q)!\,4!\,0!}\ ,$$

which is identical to the first term in the sum. The terms in the sum are similar in absolute value (and are very small). The maximum-absolute contributing term is the center term. This term is equal to

$$\frac{(-1)^{\frac{j-u}{2}}}{\left[\left(\frac{j+u}{2}+2q\right)!\right]^2\left[\left(\frac{j-u}{2}\right)!\right]^2}\ . \tag{51}$$

Due to cancellation, it is useful to consider the approximation:

$$\sum_{k'=0}^{j-u}\frac{-1^{j-u-k'}}{(j+2q-k')!\,(k'+u+2q)!\,(k')!\,(j-u-k')!} = -1^{\frac{j-u}{2}}\begin{cases}0 & ,\ j-u\ odd\\ \left[\left(\frac{j+u}{2}+2q\right)!\,\left(\frac{j-u}{2}\right)!\right]^{-2} & ,\ j-u\ even\end{cases}\ . \tag{52}$$

Substituting Eq. (52) into Eq. (49) results in

$$\begin{aligned}\hat{\lambda}_q^{(2q)}(t|\hat{\beta}_q) \approx\ &\sum_{u=0}^{x}\frac{t^u}{u!}\sum_{j=u}^{x}c_{j+2q}\\ &\times\frac{(j+u+4q)!(j+2q)!}{2^{j+2q}}\frac{-1^{\frac{j-u}{2}}\ \chi_{j-u\ even}}{\left[\left(\frac{j+u}{2}+2q\right)!\right]^2\left[\left(\frac{j-u}{2}\right)!\right]^2}\end{aligned} \tag{53}$$

where $\chi_{j-u\ even}$ is zero if $j-u$ is an odd integer and is 1 if $j-u$ is an even-valued integer. Using Stirling's approximate formula for $\ln x!$:

$$\ln x! \approx x\ln x - x\ , \tag{54}$$

some cancellations in Eq. (53) can be made. Specifically,

$$\begin{aligned}&\ln\left\{\frac{(j+u+4q)!}{2^{j+2q}}\frac{1}{\left[\left(\frac{j+u}{2}+2q\right)!\right]^2\left[\left(\frac{j-u}{2}\right)!\right]^2}\right\}\approx\\ &(j+u+4q)\ln(j+u+4q)-(j+u+4q)\ +\\ &-(j+2q)\ln 2\ +\\ &-2\left[\left(\frac{j+u}{2}+2q\right)\ln\left(\frac{j+u}{2}+2q\right)-\left(\frac{j+u}{2}+2q\right)\right]+\\ &-2\left[\left(\frac{j-u}{2}\right)\ln\left(\frac{j-u}{2}\right)-\left(\frac{j-u}{2}\right)\right]\ . \end{aligned} \tag{55}$$

After cancellations, Equation (55) results in,

$$\frac{(j+u+4q)!}{2^{j+2q}} \frac{1}{\left[\left(\frac{j+u}{2}+2q\right)!\right]^2 \left[\left(\frac{j-u}{2}\right)!\right]^2} \approx \frac{2^{j+u+4q}}{2^{j+2q}} \frac{1}{2^{u-j}(j-u)!}\,, = \frac{2^{j+2q}}{(j-u)!}\,. \tag{56}$$

Substituting Eq. (56) into Eq. (53) yields another approximation for $\hat{\lambda}_q^{(2q)}$,

$$\hat{\lambda}_q^{(2q)}(t|\hat{\beta}_q) \approx \sum_{u=0}^{x} \frac{t^u}{u!} \sum_{j=u}^{x} c_{j+2q} \frac{2^{j+2q}(j+2q)!\,(-1)^{\frac{j-u}{2}}\ \chi_{j-u\ even}}{(j-u)!} \tag{57}$$

The sums in Eq. (57) can be rearranged such that j progresses from 0 to x, and u progresses from 0 to j. Let u' equal $j-u$. Then,

$$\hat{\lambda}_q^{(2q)}(t|\hat{\beta}_q) \approx \sum_{j=0}^{x} c_{j+2q} 2^{j+2q}(j+2q)! \sum_{u'=0}^{j} \frac{t^{j-u'}}{(j-u')!\,u'!}(-1)^{\frac{u'}{2}}\ \chi_{u'\ even}\,, = 2^{2q} \sum_{j=0}^{x} c_{j+2q} \frac{(j+2q)!\,(2t)^j}{j!} \sum_{u'=0}^{j} \binom{j}{u'} t^{-u'}\,(-1)^{\frac{u'}{2}}\ \chi_{u'\ even}\,, \tag{58}$$

with the binomial coefficient $\binom{a}{b} = \frac{a!}{(a-b)!\,b!}$. The coefficients multiplying the powers of t can be collected. This results in the following representation:

$$\hat{\lambda}_q^{(2q)}(t|\hat{\beta}_q) \approx \sum_{j=0}^{x} g_j t^j\,, \tag{59}$$

for g_j specified according to Eq. (58).

Appendix D: Basic derivation of Gaussian quadrature

The polynomial, L_j, satisfies for $j' \neq j$,

$$\int_0^T L_j(t')\ L_{j'}(t')\ dt' = 0\,, \tag{60}$$

and is a Legendre polynomial. Gauss quadrature with the Legendre polynomials is sometimes referred to as "Gauss-Legendre" quadrature. The weights, w_j, $j = 1,\ \ldots,\ q$, are chosen such that the vector, $\mathbf{w} = \left[w_1\ \ldots\ w_q\right]^T$, is orthogonal to the $q-1$ vectors,

$$\mathbf{p}_k = \left[L_k(t_1)\ \ldots\ L_k(t_q)\right]^T\,, \tag{61}$$

for $k = 1,\ 2,\ \ldots\ , q-1$, with the further stipulation that

$$\mathbf{w}^T \mathbf{p}_0 = \int_0^T L_0(t')\ dt'\,. \tag{62}$$

Here $\mathbf{p}_0$ is a vector with identical entries equal to the constant L_0, and t_j, $j = 1,\ 2,\ \ldots\ , q$ are chosen as the roots of L_q:

$$L_q(t_j) = 0\,,\ j = 1,\ \ldots\ , q\,. \tag{63}$$

Thus specified, Equation (8) is exact when λ is a polynomial of order less than or equal to $2q-1$. To see this consider the integral of the order q polynomial z_{2q-1}. Following Stoer and Bulirsch (2002), this polynomial can be expressed as,

$$z_{2q-1}(t) = L_q(t)\ \tilde{q}(t) + r(t)\,, \tag{64}$$

where $\tilde{q}(t)$ and $r(t)$ are linear combinations of $L_k(t), k < q$. Then,

$$\begin{aligned} \int_0^T z_{2q-1}(t')\ dt &= \int_0^T L_q(t')\ \tilde{q}(t') + r(t')\ dt'\,, \\ &= \int_0^T r(t')\ dt'\,, && (A1) \\ &= \sum_{k=0}^{q-1} \beta_k \int_0^T L_k(t')\ dt'\,, && (Def.\ of\ r(t)) \\ &= \beta_0 \int_0^T L_0(t')\ dt'\,. && (\perp\ w.\ L_0) \end{aligned} \tag{65}$$

Similarly,

$$\begin{aligned} \sum_{j=1}^{q} w_j\ z_{2q-1}(t_j) &= \sum_{j=1}^{q} w_j \left(L_q(t_j)\ \tilde{q}(t_j) + r(t_j)\right)\,, \\ &= \sum_{j=1}^{q} w_j\ r(t_j)\,, && (L_q(t_j) = 0) \\ &= \sum_{k=0}^{q-1} \beta_k \sum_{j=1}^{q} w_j\ L_k(t_j)\,, && (\perp) \\ &= \beta_0 \int_0^T L_0(t')\ dt'\,. \end{aligned} \tag{66}$$

Here $L_q(t_j)$ is zero due to Eq. (63), and the orthogonality property of $\mathbf{w}$ is exploited. For further details see (Stoer and Bulirsch 2002) & (Press et al. 1992, §4.5).

Specification of the order of integration, q, the roots t_j and the weights w_j, $j = 1,\ \ldots\ , q$ completely specifies the Gaussian quadrature rule, Eq. (8). For the integrals approximated in this work, the t_j and weights are computed using the method specified in Golub and Welsch (1969a) for the domain of integration $(-1, 1)$. All integrals are transformed to this domain for approximation. The nodes t_j, and the weights, w_j, can be computed in a number of ways. See (Hale and Townsend 2013) for a fast alternative method

capable of accurately determining nodes and weights for Gaussian quadrature orders exceeding 100.

Appendix E: Well-behaved deviation

In the following, a sequence of lemmas are provided establishing the sense in which small quadrature error leads to small numerical error in the parameter estimates. Discussion is restricted to the univariate case ($p = 1$), without loss of generality. Let $\ell_q(\beta)$ be the log-likelihood computed with the q^{th} order Gaussian quadrature and evaluated at the parameter value β.

Lemma 1 *Concavity of ℓ_q The second derivative of $\ell_q(\beta)$ is negative for all $\beta \in \mathbb{R}$.*

Proof The proof follows from calculation:

$$\begin{aligned}\frac{d^2\ell_q(\beta)}{d\beta^2} &= \frac{d^2}{d\beta^2}\left\{\sum_{t\in T_s}\log\left(\lambda(t \mid \beta)\right) - \sum_{j=1}^{q} w_j \lambda(t_j \mid \beta)\right\}, \\ &= -\sum_{j=1}^{q} w_j \frac{d^2\lambda(t_j \mid \beta)}{d\beta^2}, \\ &= -\sum_{j=1}^{q} w_j \left[f'(\beta)\right]^2 e^{f(\beta)},\end{aligned} \tag{67}$$

for a linear differentiable function f. The weights w_j are non-negative (they are squared quantities, see for e.g. (Golub and Welsch 1969b)) guaranteeing the sign of the second derivative for $\beta \in \mathbb{R}$. □

Let $\hat{\beta}_q$ be the approximate maximum-likelihood estimate:

$$\hat{\beta}_q = \arg\max_{\beta}\ \ell_q(\beta)\,. \tag{68}$$

Let $\zeta, \zeta' \in (a, b)$ such that (9) evaluated for $\hat{\beta}$ and $\hat{\beta}_q$, is, respectively, $\delta_{\hat{\beta}}$, and $\delta_{\hat{\beta}_q}$:

$$\delta_{\hat{\beta}} = \left|\frac{\lambda^{(2q)}(\zeta|\hat{\beta})}{(2q)!\,k_q^2}\right|, \tag{69}$$

$$\delta_{\hat{\beta}_q} = \left|\frac{\lambda^{(2q)}(\zeta'|\hat{\beta}_q)}{(2q)!\,k_q^2}\right|. \tag{70}$$

Then there exists a δ for any quadrature order q,

$$\delta = \max\left\{\delta_{\hat{\beta}}, \delta_{\hat{\beta}_q}\right\} \tag{71}$$

such that

$$\left|\ell(\beta) - \ell_q(\beta)\right| < \delta\,,\ \beta \in \left\{\hat{\beta}, \hat{\beta}_q\right\}\,. \tag{72}$$

The following lemma can be proven.

Lemma 2 *Log-Likelihood Approximation*

$$\left|\ell(\hat{\beta}) - \ell_q(\hat{\beta}_q)\right| < \delta\,. \tag{73}$$

Proof Suppose, $\ell_q(\hat{\beta}_q)$ is less than $\ell(\hat{\beta}_q)$. By Eq. (72)

$$\ell(\hat{\beta}_q) < \ell_q(\hat{\beta}_q) + \delta\,. \tag{74}$$

From Eq. (74) and concavity the smallest that $\ell(\hat{\beta})$ can be is $\ell(\hat{\beta}_q)$. Then $\ell_q(\hat{\beta}_q) - \ell(\hat{\beta}) < \delta$. Similarly, by concavity, $\ell_q(\hat{\beta}) + \delta$ is less than $\ell_q(\beta_q) + \delta$. Then by (72) $\ell(\hat{\beta})$ is upper-bounded:

$$\ell(\hat{\beta}) \le \ell_q(\hat{\beta}) + \delta \le \ell_q(\hat{\beta}_q) + \delta\,, \tag{75}$$

again implying $\ell(\hat{\beta}) - \ell_q(\hat{\beta}_q) \le \delta$.

If instead $\ell_q(\hat{\beta}_q) > \ell(\hat{\beta}_q)$, then

$$\ell_q(\hat{\beta}_q) < \ell(\hat{\beta}_q) + \delta\,. \tag{76}$$

By concavity we have,

$$\ell_q(\hat{\beta}) \le \ell_q(\hat{\beta}_q) < \ell(\hat{\beta}_q) + \delta \le \ell(\hat{\beta}) + \delta\,. \tag{77}$$

From Eq. (72) $|\ell_q(\hat{\beta}) - \ell(\hat{\beta})| < \delta$ implying $|\ell_q(\hat{\beta}_q) - \ell(\hat{\beta})| < \delta$, and the proof is complete. □

Having established the proximity between the log-likelihood at $\hat{\beta}$ with the approximate log-likelihood at $\hat{\beta}_q$, it remains to show that $\hat{\beta}_q$ approximates $\hat{\beta}$.

Lemma 3 *$\hat{\beta}_q$ approximates $\hat{\beta}$*
Fix $\delta > 0$. If

$$\left|\ell(\hat{\beta}) - \ell_q(\hat{\beta}_q)\right| < \delta\,, \tag{78}$$

then there exists an $\epsilon > 0$,

$$\left|\hat{\beta}_q - \hat{\beta}\right| < \epsilon\,, \tag{79}$$

such that

$$\left|\epsilon^2 \frac{d^2\ell_q(\hat{\beta}_q)}{2\,d\beta^2} + \epsilon^3 \frac{d^3\ell_q(\eta)}{6\,d\beta^3}\right| < 2\delta\,. \tag{80}$$

Proof Taylor expanding ℓ_q about $\hat{\beta}_q$ and evaluating at $\hat{\beta}$ yields:

$$\begin{aligned}\ell_q(\hat{\beta}) = \ell_q(\hat{\beta}_q) &+ \frac{d^2\ell_q(\hat{\beta}_q)}{2\,d\beta^2}\left(\hat{\beta} - \hat{\beta}_q\right)^2 \\ &+ \frac{d^3\ell_q(\eta)}{6\,d\beta^3}\left(\hat{\beta} - \hat{\beta}_q\right)^3\end{aligned} \tag{81}$$

with $\eta \in \left(\hat{\beta}_q, \hat{\beta}\right)$. By the triangle inequality,

$$\begin{aligned}\left|\ell_q(\hat{\beta}) - \ell_q(\hat{\beta}_q)\right| &\leq \left|\ell_q(\hat{\beta}) - \ell(\hat{\beta})\right| + \left|\ell(\hat{\beta}) - \ell_q(\hat{\beta}_q)\right| , \\ &< \delta + \delta , \\ &= 2\delta . \end{aligned} \tag{82}$$

Then:

$$\left|\epsilon^2 \frac{d^2\ell_q(\hat{\beta}_q)}{2\, d\beta^2} + \epsilon^3 \frac{d^3\ell_q(\eta)}{6\, d\beta^3}\right| < 2\delta , \tag{83}$$

with $\epsilon = \hat{\beta} - \hat{\beta}_q$. □

Appendix F: Maximum parameter estimate error

The δ in Lemma 3 is the larger of the two quadrature errors, $\delta_{\hat{\beta}_q}$ and $\delta_{\hat{\beta}}$; the former computed for the known $\hat{\beta}_q$, and the other for the unknowable $\hat{\beta}$. If a bound is placed upon $\lambda^{(2q)}/k_q^2$, and the limit taken as q tends to infinity, both of these error bounds tend to zero, and hence become close. Here, to obtain an estimate of ϵ it is assumed that $\delta = \delta_{\hat{\beta}_q} = \delta_{\hat{\beta}}$. With this specification, ϵ, the parameter estimate deviation from the true MLE can be specified. From Eq. (9), set

$$\left|\frac{\lambda^{(2q)}(\zeta \mid \hat{\beta}_q)}{(2q)!\, k_q^2}\right| = \delta . \tag{84}$$

Then, for η as specified after Eq. (81),

$$\left|\epsilon^2 \frac{d^2\ell_q(\hat{\beta}_q)}{2d\beta^2} + \epsilon^3 \frac{d^3\ell_q(\eta)}{6\, d\beta^3}\right| < 2\left|\frac{\lambda^{(2q)}(\zeta \mid \hat{\beta}_q)}{(2q)!\, k_q^2}\right| . \tag{85}$$

With this specification,

$$\epsilon^2 < -4\left(\frac{d^2\ell_q(\hat{\beta}_q)}{d\beta^2}\right)^{-1}\left|\frac{\lambda^{(2q)}(\zeta \mid \hat{\beta}_q)}{(2q)!\, k_q^2}\right| + O(\epsilon^3) . \tag{86}$$

It is useful to set ϵ^2 equal to the bound in Eq. (86). Let

$$\sigma_\beta = \left|\frac{d^2\ell_q(\hat{\beta}_q)}{d\beta^2}\right|^{-1/2} , \tag{87}$$

and

$$x = 2\sigma_\beta \sqrt{\frac{\left|\lambda^{(2q)}(\zeta \mid \hat{\beta}_q)\right|}{(2q)!\, k_q^2}} . \tag{88}$$

Then,

$$\epsilon = x + O\left(\epsilon^3\right) . \tag{89}$$

Appendix G: Accuracy of observed fisher information

Lemma 4 *ϵ-Equivalence of Observed Fisher Information*

Let ϵ be as specified in Eq. (89). Introduce the unbiased estimators $\tilde{\beta}$, $\tilde{\beta}_q$, whose realizations are the estimates $\hat{\beta}$ and $\hat{\beta}_q$ of the parameters β and β_q. Further, let the realization of the random variable X be the quadrature error for any given data set, and specify X to be independent of the true MLE estimator $\tilde{\beta}$. Then the variance, var $\left\{\tilde{\beta}_q\right\}$ satisfies:

$$\left|var\left\{\tilde{\beta}_q\right\} - var\left\{\tilde{\beta}\right\}\right| \leq 4\epsilon^2 . \tag{90}$$

Proof The proof follows from direct calculation. Consider

$$\begin{aligned} var\left\{\tilde{\beta}_q\right\} &= E\left\{\left(\tilde{\beta}_q - \beta_q\right)^2\right\} , \\ &= E\left\{\left[\left(\tilde{\beta} + X\right) - \beta_q\right]^2\right\} , \\ &= var\left\{\tilde{\beta}\right\} + \left[\beta^2 + \beta_q^2 - 2\beta_q\beta\right] \\ &\quad +2\, E\left\{X\tilde{\beta}\right\} - 2\beta_q E\{X\} + E\left\{X^2\right\} . \end{aligned} \tag{91}$$

For some realized quadrature error η, $|\eta| < \epsilon$, the term,

$$\beta^2 + \beta_q^2 - 2\beta_q\beta = \beta^2 + (\beta + \eta)^2 - 2(\beta + \eta)\beta , \tag{92}$$

$$= \eta^2 . \tag{93}$$

Then

$$\beta^2 + \beta_q^2 - 2\beta_q\beta \leq \epsilon^2 . \tag{94}$$

Similarly the contribution to Eq. (91) from the terms involving X can be bounded:

$$\begin{aligned} E\left\{X^2\right\} + 2E\{X\}\left(\beta - \beta_q\right) &= E\left\{X^2\right\} - 2\eta E\{X\} , \\ &\leq E\left\{X^2\right\} + 2\left|\eta E\{X\}\right| , \\ &\leq \epsilon^2 + 2\epsilon^2 . \end{aligned} \tag{95}$$

Equations (91), (94), and (95) imply Eq. (90). □

Acknowledgments The authors thank Howard Eichenbaum for his support. Thanks goes to Robert E. Kass for a discussion regarding the content of this paper and on the use of Gaussian quadrature in statistics, to Mikio Aoi for a useful comment regarding the scope of the paper, and to Sujith Vijayan for a useful discussion regarding the neural action potential and refractory effect. KQL is supported by NSF grant DMS-1042134.

References

Barbieri, R., Frank, L.M., Nguyen, D.P., Quirk, M.C., Solo, V., Wilson, M.A., & Brown, E.N. (2004). Dynamic analyses of information encoding in neural ensembles. *Neural Computation*, *16* (2), 277–307.

Brown, E., Barbieri, R., Ventura, V., Kass, R., & Frank, L. (2002). The time-rescaling theorem and its application to neural spike train data analysis. *Neural computation*, *14*(2), 325–346.

Citi, L., Ba, D., Brown, E.N., & Barbieri, R. (2014). Likelihood methods for point processes with refractoriness. *Neural Computation*, *26*(2), 237–263.

Daley, D.J., & Vere-Jones, D. (2003). *An introduction to the theory of point processes*: Springer Series in Statistics.

Davis, P.J., & Rabinowitz, P. (1967). *Numerical integration*: Blaisdell Publishing Company London.

Genz, A., & Kass, R.E. (1991). An application of subregion adaptive numerical integration to a bayesian inference problem. *Computing Science and Statistics*, *23*, 441–444.

Genz, A., & Kass, R.E. (1997). Subregion-adaptive integration of functions having a dominant peak. *Journal of Computational and Graphical Statistics*, *6*(1), 92–111.

Golub, G.H., & Welsch, J.H. (1969). Calculation of gauss quadrature rules. *Mathematics of Computation*, *23*(106), 221–230+s1–s10.

Golub, G.H., & Welsch, J.H. (1969). Calculation of gauss quadrature rules. *Mathematics of Computation*, *23*(106), 221–230.

Hale, N., & Townsend, A. (2013). Fast and accurate computation of gauss–legendre and gauss–jacobi quadrature nodes and weights. *SIAM Journal on Scientific Computing*, *35*(2), A652—A674.

Henze, D., & Buzsaki, G. (2001). Action potential threshold of hippocampal pyramidal cells in vivo is increased by recent spiking activity. *Neuroscience*, *105*(1), 121–130.

Kass, R.E., Ventura, V., & Cai, C. (2003). Statistical smoothing of neuronal data. *Network-Computation in Neural Systems*, *14* (1), 5–16.

Kesner, R.P., Hunsaker, M.R., & Gilbert, P.E. (2005). The role of ca1 in the acquisition of an object-trace-odor paired associate task. *Behavioral Neuroscience*, *119*(3), 781–786.

Kuonen, D. (2003). Numerical integration in s-plus or r: A survey. *Journal of Statistical Software*, *8*(13), 1–14.

Lepage, K.Q., Gregoriou, G.G., Kramer, M.A., Aoi, M., Gotts, S.J., Eden, U.T., & Desimone, R. (2013). A procedure for testing across-condition rhythmic spike-field association change. *Journal of neuroscience methods*, *213*(1), 43–62.

Lepage, K.Q., MacDonald, C.J., Eichenbaum, H., & Eden, U.T. (2012). The statistical analysis of partially confounded covariates important to neural spiking. *Journal of neuroscience methods*, *205*(2), 295–304.

MacDonald, C., Lepage, K., Eden, U., & Eichenbaum, H. (2011). Hippocampal "time cells" bridge the gap in memory for discontiguous events. *Neuron*, *71*(4).

McCullagh, P., & Nelder, J.A. (1999). *Generalized Linear Models*, 2nd: Chapman & Hall/CRC.

Mena, G., & Paninski, L. (2014). *On quadrature methods for refractory point process likelihoods*: Neural Computation. In press.

Paninski, L. (2004). Maximum likelihood estimation of cascade point-process neural encoding models. *Network: Computation in Neural Systems*, *15*(4), 243–262.

Paninski, L., Ahmadian, Y., Ferreira, D.G., Koyama, S., Rad, K.R., Vidne, M., Vogelstein, J., & Wu, W. (2010). A new look at state-space models for neural data. *Journal of Computational Neuroscience*, *29*(1-2), 107–126.

Press, W.H., Teukolsky, S.A., Vetterling, W.T., & Flannery, B.P. (1992). *Numerical recipes in C (2nd ed.): the art of scientific computing*. NY, USA: Cambridge University Press.

Ramirez, A.A., & Paninski, L. (2013). *Fast generalized linear model estimation via expected log-likelihoods*: Journal of Computational Neuroscience, In press.

Shewchuk, J.R. (1994). An introduction to the conjugate gradient method without the agonizing pain.

Snyder, D.L. (1975). Random point processes.

Stevenson, I.H., & Kording, K.P. (2011). How advances in neural recording affect data analysis. *Nature neuroscience*, *14*(2), 139–142.

Stoer, J., & Bulirsch, R. (2002). *Introduction to numerical analysis*, 3rd, Vol. 12: Springer.

Trefethen, L.N. (2008). Is gauss quadrature better than clenshaw-curtis *SIAM Review*, *50*(1), 67–87.

Truccolo, W., Eden, U.T., Fellows, M.R., Donoghue, J.P., & Brown, E.N. (2005). A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. *Journal Neurophysiology*, *93*(2), 1074–1089.