Fast Methods Comp

Joshua T. Vogelstein, Baktash Babidi, Liam Paninski

May 8, 2008

Contents

Abstract

1 Introduction

The goal of this work is to infer the underlying spike train, having only the observed fluorescence signal available to us. To do so, we propose a parametric model of the experimental system. Then, we develop a number of algorithms, each iteratively estimating the parameters and then using those parameters to infer the hidden spike trains.

2 Methods

2.1 Model

First, for simplicity, we assume that we have extracted a time series of fluorescence observations, F_t, which is a function of the intracellular calcium concentration of the imaged soma at that time, $[\text{Ca}^{2+}]_t$. In particular, we assume that F_t simply a noisy version of $[\text{Ca}^{2+}]_t$, i.e.,

$$F_t = [\text{Ca}^{2+}]_t + \varepsilon_t, \qquad \varepsilon_t \sim \mathcal{N}(0, \sigma^2), \tag{1}$$

where ε_t is an additive Gaussian noise term with zeromean and variance σ^2. Note that this time series is naturally discrete, with time step size Δ corresponding to the frame rate of the image acquisition system, with t indexing the time step, and T indicating the final time step. The $[\text{Ca}^{2+}]$ signal is assumed to jump immediate after each spike by A μM and then decay back to zero, with time constant τ sec. Therefore, we have:

$$[\text{Ca}^{2+}]_t = (1 - \frac{\Delta}{\tau})[\text{Ca}^{2+}]_{t-1} + An_t, \tag{2}$$

where n_t indicates the number of spikes at each time step, and is given by an exponential distribution with rate λ_t:

$\text{P}(\text{n}_t|\lambda_t) = \lambda_t e^{-\lambda_t n_t \Delta}$.

The firing rate may be any function of the stimulus, but to ensure that the likelihood of the parameters have only a single peak, we typically choose a Generalized Linear Model **??**. Figure 1 shows an example fluorescence time-series (A), the true calcium signal (B), and the true spike train (C), the expected firing rate (D) , and the stimulus (E).

2.2 Inferring the spike times

We are interested in finding the spike train that caused the observed fluorescence time series. Formally, we may be interested in computing the most likely spike train, conditioned on the fluorescence data:

$$\begin{aligned}\widehat{\boldsymbol{n}} &= \underset{\boldsymbol{n}}{\operatorname{argmax}}\, P(\boldsymbol{n}|\boldsymbol{F}) = \underset{\boldsymbol{n}}{\operatorname{argmax}}\, P(\boldsymbol{F}|\boldsymbol{n})P_{\boldsymbol{\theta}}(\boldsymbol{n}) \\ &= \underset{\boldsymbol{n}}{\operatorname{argmax}} \prod_t P_{\boldsymbol{\theta}}(F_t|n_t)P(n_t) = \underset{\boldsymbol{n}}{\operatorname{argmax}} \sum_t \log P(F_t|n_t) + \log P(n_t) \\ &= \underset{\boldsymbol{n}}{\operatorname{argmin}} \sum_t c(F_t - a[\mathrm{Ca}^{2+}]_{t-1} - An_t)^2 + \lambda_t n_t \Delta, \end{aligned} \tag{3}$$

where we let $c = 1/(2\sigma^2)$ and $a = (1 - \Delta/\tau)$. This is a typical regularized quadratic optimization problem, which may be solved quickly and analytically (see Appendix **??** for details). Note that this is equivalent to a Weiner filter. However, when data is noisy, this solution tends to fit the noise as signal (see Figure 2 (B)). As such, it is desirable to impose certain constraints on the system. In particular, since we know that spike trains are non-negative entities, we could instead solve:

$$\boldsymbol{n}_{nng} = \underset{\boldsymbol{n}:n_t \geq 0}{\operatorname{argmin}} \sum_t c(F_t - a[\mathrm{Ca}^{2+}]_{t-1} - An_t)^2 + \lambda_t n_t \Delta, \tag{4}$$

where the only difference between (3)and (4) is that we've added the constraint that at each time step n_t must be non-negative. Alternately, we could require that each n_t is an integer:

$$\boldsymbol{n}_{int} = \underset{\boldsymbol{n}:n_t \in 0,1,2,\ldots}{\operatorname{argmin}} \sum_t c(F_t - a[\mathrm{Ca}^{2+}]_{t-1} - An_t)^2 + \lambda_t n_t \Delta. \tag{5}$$

In the next subsections, we provide an efficient algorithm for solving (4) and (5).

2.2.1 Barrier Method

To efficiently solve this problem, we adopt the "barrier method". This approach iteratively solves a related problem that penalizes the algorithm for inferring negative values, but with each iteration, reduces the penalty, in such a way that is guaranteed to converge to (4)**??**:

$$\boldsymbol{n}_{\eta} = \underset{\boldsymbol{n}}{\operatorname{argmin}} \sum_{t=0}^{T} c(F_t - a[\mathrm{Ca}^{2+}]_{t-1} - An_t)^2 + \lambda_t n_t - \eta \log n_t, \tag{6}$$

where the "barrier term" $-\log(n_t)$ approaches infinity as n_t approaches zero. By recursively solving (6), reducing η each time, $\boldsymbol{n}_\eta \rightarrow \boldsymbol{n}_{nng}$. While this function is no longer quadratic, it is still concave. Therefore, the solution may be found be iterating any gradient descent technique. For instance, one can use Newton's method to iteratively increase the likelihood. Importantly, the spike train may be written as a linear function of the calcium:

$$\boldsymbol{n} = \boldsymbol{M}\boldsymbol{C}$$

$$\begin{bmatrix} n_0 \\ n_1 \\ \vdots \\ n_T \end{bmatrix} = \begin{bmatrix} -a/A & 0 & 0 & \cdots \\ 1/A & -a/A & 0 & \cdots \\ \vdots & & & \\ 0 & \cdots & 1/A & -a/A \end{bmatrix} \begin{bmatrix} [\mathrm{Ca}^{2+}]_0 \\ [\mathrm{Ca}^{2+}]_1 \\ \vdots \\ [\mathrm{Ca}^{2+}]_T \end{bmatrix} \tag{7}$$

where $\boldsymbol{M}$ is a bidiagonal matrix. Therefore, (6) may be solved in terms of $\boldsymbol{C} = \{[\mathrm{Ca}^{2+}]_0, \ldots, [\mathrm{Ca}^{2+}]_T\}$, by iteratively solving:

$$\boldsymbol{C}_i = \boldsymbol{C}_{i-1} + s\boldsymbol{d}, \qquad \boldsymbol{d} = \boldsymbol{H}^{-1}\boldsymbol{g} \tag{8}$$

where s is the step size, d is the step direction, and i indexes the iteration. s must be chosen such that all of our constraints, namely $n_t > 0$ for all t, are satisfied:

$$0 < n_t = \boldsymbol{M}(\boldsymbol{C} + s\boldsymbol{d}) \Rightarrow s > -\boldsymbol{M}\boldsymbol{C}(\boldsymbol{M}\boldsymbol{d})^{-1}. \tag{9}$$

Furthermore, it is possible to "over-step", or rather, take a step so large as to *increase* the likelihood that we are trying to minimize. As such, prior to stepping, we check that the likelihood would indeed decrease, if not, we reduce s until it does.

To compute $\boldsymbol{d}$, we must solve for the Hessian and gradient of our likelihood function with respect to $\boldsymbol{C}$. Writing the likelihood in vector notation, we can solve for both:

$$\mathcal{L} = c\left\|\boldsymbol{F} - \boldsymbol{C}\right\|_2^2 + \boldsymbol{\lambda}\boldsymbol{M}\boldsymbol{C} - \eta\log(\boldsymbol{M}\boldsymbol{C}) \tag{10}$$

$$\boldsymbol{g} = -2c(\boldsymbol{F} - \boldsymbol{C}) + \boldsymbol{\lambda}\boldsymbol{M} - \eta\boldsymbol{M}'(\boldsymbol{M}\boldsymbol{C})^{-1} \tag{11}$$

$$\boldsymbol{H} = 2c\boldsymbol{I} + 2\eta\boldsymbol{M}'(\boldsymbol{M}\boldsymbol{C})^{-2}\boldsymbol{M}. \tag{12}$$

Plugging $\boldsymbol{g}$ and $\boldsymbol{H}$ into (8) and then choosing the step size s that maximizes this likelihood, but also satisfies (9) and decrease the likelihood, yields $\boldsymbol{C}_i$. This is iterated until it converges, at which time η is reduced and the whole process repeats. Importantly, the Hessian here is tridiagonal (which follows from the fact that $\boldsymbol{M}$ is bidiagonal), facilitating $O(T)$ time to solve each Newton step, as opposed to $O(T^3)$, which would be unacceptably long for large data sets. Pseudocode for an efficient implementation of this approach is provided in Algorithm 1. Figure 1(C) shows how this constraint further improves the inference accuracy.

Algorithm 1 Pseudocode for the log barrier method

while $\eta > \epsilon$ **do**
  Initialize $\boldsymbol{C}_i$
  $\mathcal{L}_i = c\,\|\boldsymbol{F} - \boldsymbol{C}_i\|_2^2 + \boldsymbol{\lambda}\boldsymbol{M}\boldsymbol{C}_i - \eta \log(\boldsymbol{M}\boldsymbol{C}_i)$
  while $\mathcal{L}_i < \mathcal{L}_{i-1}$ **do**
    $\boldsymbol{g} = -2c(\boldsymbol{F} - \boldsymbol{C}_i) + \boldsymbol{\lambda}\boldsymbol{M} - \eta\boldsymbol{M}'(\boldsymbol{M}\boldsymbol{C}_i)^{-1}$
    $\boldsymbol{H} = 2c\boldsymbol{I} + 2\eta\boldsymbol{M}'(\boldsymbol{M}\boldsymbol{C}_i)^{-2}\boldsymbol{M}$
    $\boldsymbol{d} = \boldsymbol{H}^{-1}\boldsymbol{g}$ efficiently
    Choose s such that
      $s > -\boldsymbol{M}\boldsymbol{C}_i(\boldsymbol{M}\boldsymbol{d})^{-1}$
      and $\mathcal{L}_i < \mathcal{L}_{i-1}$
    Let $\boldsymbol{C}_i = \boldsymbol{C}_{i-1} + s\boldsymbol{d}$
    $i \leftarrow i + 1$
  end while
  reduce η
end while

Barrier Method for Intermittent data The barrier method may be generalized in several interesting ways. For example, sometimes the fluorescence time-series is "missing" some data point due to camera mishaps, or stimulus artifacts, for instance. Alternately, if the precision of inferrence is higher than the frame rate, one can consider only some subset of time steps observed, and the rest missing. In either case, one can reformulate the likelihood, gradient, and Hessian for this scenario as:

$$\mathcal{L} = \sum_{t=0}^{T} ([\text{Ca}^{2+}]_t - a[\text{Ca}^{2+}]_{t-1})^2 + f(n_t) + z_t(F_t - [\text{Ca}^{2+}]_t)^2$$

$$= \|\boldsymbol{C}_t - a\boldsymbol{C}_{t-1}\|_2^2 + f(\boldsymbol{n}) + \big(\boldsymbol{z}(\boldsymbol{F} - \boldsymbol{C})\big)^2 \tag{13}$$

$$\boldsymbol{g} = 2(\boldsymbol{C}_t - a\boldsymbol{C}_{t-1}) + \nabla_c f(\boldsymbol{n}) + 2\boldsymbol{D}(\boldsymbol{z})(\boldsymbol{F} - \boldsymbol{C}) \tag{14}$$

$$\boldsymbol{H} = 2\boldsymbol{I} + \nabla_c^2 f(\boldsymbol{n}) + 2\boldsymbol{D}(\boldsymbol{z}) \tag{15}$$

where $\boldsymbol{z} = [z_0, z_T]$ takes a value 1 during observation frames and 0 otherwisei, and for brevity we write the log-barrier term and the regularizing term as $f(n_t)$, and $\boldsymbol{D}(\boldsymbol{z})$ indicates a matrix with $\boldsymbol{z}$ along the diagonal, and zero elsewhere.

2.2.2 Projection Pursuit Regression (PPR)

Although the barrier method provides us with a meaningful spike train inference, we can further constrain the inferred spike train to be integers, as in (5). Unforunately, solving this problem exactly is known to be NP-hard**??**, as there are 2^T possible solutions. Therefore, we instead find an approximate solution using an algorithm called *projection pursuit regression* (PPR)**??** (see Algorithm 2 for pseudocode for this problem). Figure 1(D) shows the inferred spike train using this algorithm. Although this approach imposes a desirable constraint (i.e., that spike trains include only integers), it also comes at a cost relative to the barrier method, as will be described in Section 3.

Algorithm 2 Pseudocode for PPR

Initialize $r_t^{(0)} = F_t, \epsilon_0 = \sum_t (r_t^{(0)})^2$
while $\epsilon_i < \epsilon_{i-1}$ **do**
  $t^{(i+1)} = \max_t r^{(i)} \otimes Ae^{-t/\tau} + \lambda_t$
  Let $\boldsymbol{o}^{(i+1)}$ be a zero vector with unity at $t^{(i+1)}$
  $[\text{Ca}^{2+}]_t^{(i+1)} = o_t^{(i+1)} * Ae^{-t/\tau}$
  $r_t^{(i+1)} = r_t^{(i)} - [\text{Ca}^{2+}]_{i+1}$
  $n_t^{(i+1)} = n_t^{(i)} + o_t^{(i+1)}$
  $\epsilon_{i+1} = \sum_t \left(r_t^{(i+1)}\right)^2 + \lambda_t n_t^{(i+1)}$
end while

2.3 Inferring the parameters

All of the above approaches assume a particular parametric model. As such, these parameters must be learned. Formally, we would like to compute:

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \int P_{\boldsymbol{\theta}}(\boldsymbol{F}|\boldsymbol{n}) P_{\boldsymbol{\theta}}(\boldsymbol{n}) d\boldsymbol{n} \tag{16}$$

$$\{\widehat{A}, \widehat{a}, \widehat{c}, \widehat{\boldsymbol{\lambda}}\} = \underset{A,a,c,\boldsymbol{\lambda}>0}{\operatorname{argmax}} \sum_{t=0}^{T} \int c(F_t - a[\mathrm{Ca}^{2+}]_{t-1} - An_t)^2 - \lambda_t n_t \Delta d n_t \tag{17}$$

where $\boldsymbol{\theta} = \{A, a, c, \boldsymbol{\lambda}\}$. However, since we cannot solve the above stochastic integrals, we instead approximate (16) with:

$$\{\widehat{A}, \widehat{a}, \widehat{c}, \widehat{\boldsymbol{\lambda}}\} = \underset{A,a,c,\boldsymbol{\lambda}>0}{\operatorname{argmax}} \sum_{t=0}^{T} c(F_t - a[\mathrm{Ca}^{2+}]_{t-1} - A\widehat{n}_t)^2 - \lambda_t \widehat{n}_t \Delta \tag{18}$$

where $\widehat{\boldsymbol{n}}$ is the solution obtained from solving either (4) or (5). This seems as if it should be a good approximation because the likelihood function is very high dimensional, and only a few sequences are actually likely. (18) separates out into a number of concave problems. This may be seen by making the following subsitutions:

$$\boldsymbol{C}_{est} = \begin{bmatrix} [\widehat{\mathrm{Ca}}^{2+}]_1 \\ \vdots \\ [\widehat{\mathrm{Ca}}^{2+}]_T \end{bmatrix}, \boldsymbol{n}_{est} = \begin{bmatrix} \widehat{n}_0 \\ \vdots \\ \widehat{n}_{T-1} \end{bmatrix}, \boldsymbol{\lambda}_{est} = \begin{bmatrix} \lambda_0 \\ \vdots \\ \lambda_{T-1} \end{bmatrix},$$

$$\boldsymbol{F} = \begin{bmatrix} F_0 \\ \vdots \\ F_{T-1} \end{bmatrix}, \boldsymbol{G} = -\begin{bmatrix} \boldsymbol{C}_{est} \\ \boldsymbol{n}_{est} \end{bmatrix}', \quad \boldsymbol{x} = \begin{bmatrix} a \\ A \end{bmatrix},$$

and plugging them into (18):

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}>\mathbf{0}}{\operatorname{argmin}}\, c \left\| \boldsymbol{G}\boldsymbol{x} + \boldsymbol{F} \right\|_2^2 + \boldsymbol{\lambda}'\boldsymbol{n} \tag{19}$$

$$= \underset{\boldsymbol{x}>\mathbf{0}}{\operatorname{argmin}} \frac{1}{T} (\boldsymbol{x}'\boldsymbol{G}'\boldsymbol{G}\boldsymbol{x} + 2\boldsymbol{G}'\boldsymbol{F}\boldsymbol{x} \tag{20}$$

Therefore, sovling for $\widehat{\boldsymbol{x}}$ is a constrained quadratic optimization problem where $\boldsymbol{G}'\boldsymbol{G}$ is the quadratic term and $\boldsymbol{G}'\boldsymbol{F}$ is the linear term. c may be solved for analytically:

$$\sigma^2 = \frac{1}{T}\sum_{t=0}^{T}(F_t - [\widehat{\mathrm{Ca}}^{2+}]_t)^2 \tag{21}$$

where $[\widehat{\mathrm{Ca}}^{2+}]_t$ is the nferred $[\mathrm{Ca}^{2+}]$ from either (4) or (5). Finally, $\boldsymbol{\lambda}$ may be computed by using:

$$\lambda_t = \frac{1}{\widehat{A}\widehat{n}_t\Delta} \tag{22}$$

3 Results

Figure 1(E) shows the output of the algorithm proposed previously by Yaksi and Friedrich (2006)**??**, which may be thought of as a custom approximation to either of these regularization approaches, and

4 Figures

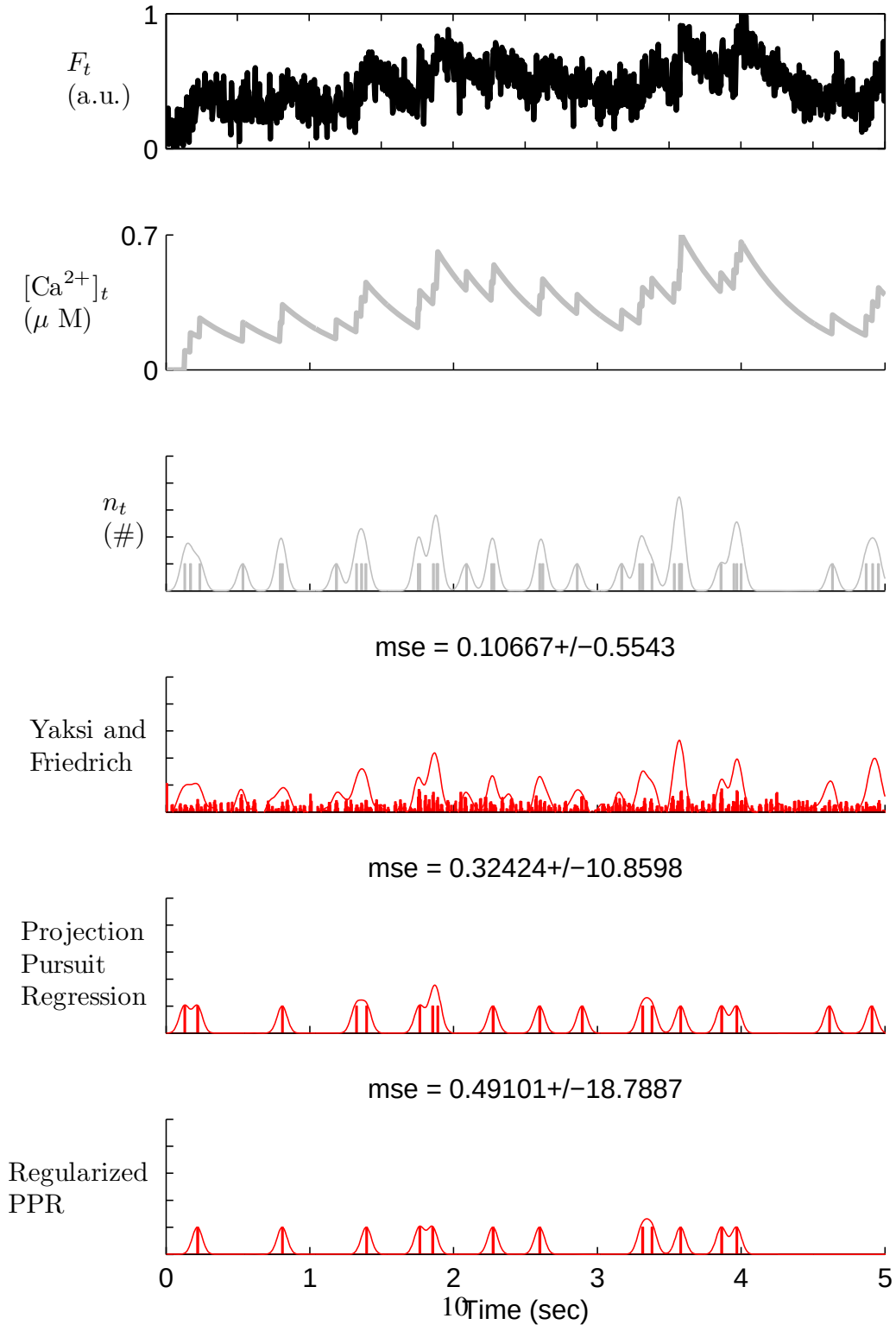
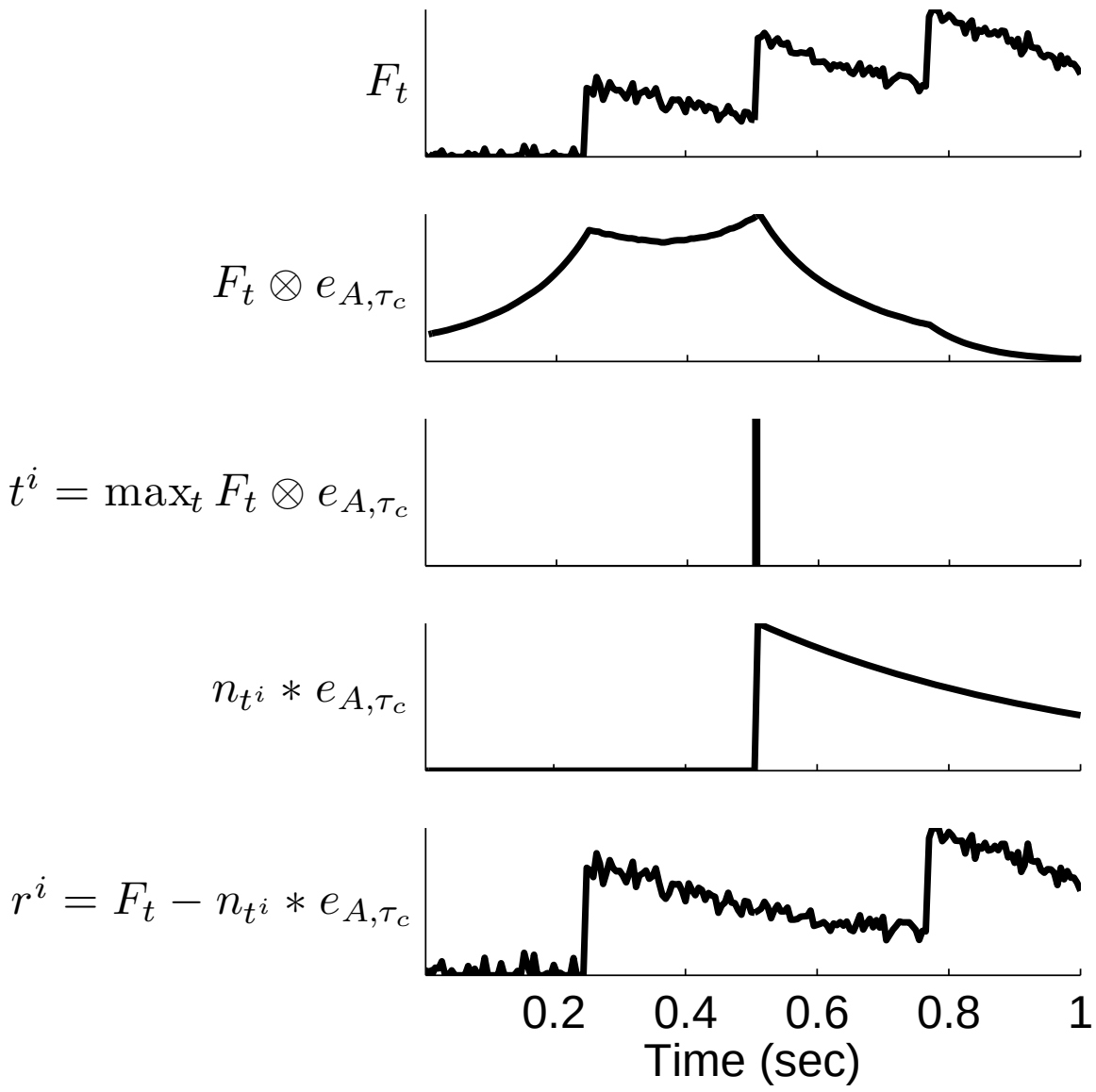


Figure 1: Comparison of various methods of inferring spike times.


F_t
$F_t \otimes e_{A,\tau_c}$
$t^i = \max_t F_t \otimes e_{A,\tau_c}$
$n_{t^i} * e_{A,\tau_c}$
$r^i = F_t - n_{t^i} * e_{A,\tau_c}$
0.2
0.4
0.6
0.8
1
Time (sec)