Online non-negative deconvolution for spike train inference from population calcium imaging

Joshua T. Vogelstein, Adam M. Packer, Tim A. Machado,
Tanya Sippy, Baktash Babadi, Rafael Yuste, Liam Paninski

November 29, 2009

Abstract

Calcium imaging technologies for observing spiking activity from large populations of neurons are quickly gaining popularity. While the raw data are fluorescence movies often the underlying spike trains are of interest. This work presents an online non-negative deconvolution filter to infer the approximately most likely spike train for each neuron, given the fluorescence observations. This algorithm outperforms optimal linear deconvolution (aka, Wiener filter) on both simulated and in vitro data. The performance gains come from restricting the inferred spike trains to be positive (using an interior-point method), unlike the Wiener filter. The algorithm is fast enough that even when imaging ≈ 100 neurons, inference can be performed on the set of all observed traces in super-real-time (ie, faster than real-time). Performing optimal spatial filtering on the images further refines the estimates. Importantly, all the parameters required to perform the inference can be estimated using only the fluorescence data, obviating the need to perform joint electrophysiological and imaging calibration experiments.

1 Introduction

Simultaneously imaging large populations of neurons using calcium sensors is becoming increasingly popular [1], both in vitro [2, 3] and in vivo [4, 5, 6], and is expected to continue as the signal-to-noise-ratio (SNR) of genetic sensors continues to improve [7, 8, 9]. Whereas the data from these experiments are movies of time-varying fluorescence traces, the desired signal is typically the spike trains of the observable neurons. Unfortunately, finding the most likely spike train is a challenging computational task, due to poor SNR, poor temporal resolution, unknown parameters, and computational intractability. One is therefore effectively forced to find an approximately most likely spike train, or hope that the inferred spike train is indeed most likely.

A number of groups have therefore proposed algorithms to infer spike trains from calcium fluorescence data. For instance, Greenberg et al. [10] developed a novel template matching algorithm. Holekamp et al. [11] took a very different strategy, by performing the optimal linear deconvolution (ie, the Wiener filter) on the fluorescence data. This approach is natural from a signal processing standpoint, but does not utilize the knowledge that spikes are always positive. Vogelstein et al. [12] proposed a sequential Monte Carlo method to efficiently estimate the probability of a spike in each image frame, given the entire fluorescence time-series. While effective, that approach is not suitable for online analyses of populations of neurons, as the computations run in about real-time per neuron (ie, analyzing one minute of data requires about one minute of computational time on a standard laptop computer).

The present work starts by building a generative model relating spiking activity to fluorescence traces. Unfortunately, inferring the most likely spike train given this model is computationally intractable. Making some reasonable approximations leads to an algorithm that infers the approximately most likely spike train, given the fluorescence data. This algorithm has a few particularly noteworthy features, relative to other approaches. First, spikes are assumed to be positive. This assumption often improves filtering results when the underlying signal has this property [13, 14, 15, 16, 17, 18, 19, 20]. Second, the algorithm is extremely fast: it can process a calcium trace from 50,000 images in about one second on a standard laptop computer. In fact, filtering the signals for an entire population of about 100 neurons runs in super-real-time (ie, *faster* than real-time). This speed facilitates using this filter online, as observations are being collected. In addition to these two features, the model may be generalized in a number of ways, including incorporating spatial filtering of the raw movie. The efficacy of the proposed filter is demonstrated on several real data-sets, suggesting this algorithm is a powerful and robust tool for online spike train inference. The code (which is a simple Matlab script) is available from the authors upon request.

2 Methods

2.1 Data driven generative model

Figure 1 shows data from a typical in vitro, epifluorescence experiment (see section 2.7 for data collection details). The top panel shows the mean frame of this movie, including 3 neurons, two of which are patched. To build the model, the pixels within a region-of-interest (ROI) are selected (white circle). Given the ROI, all the pixel intensities of each frame can be averaged, to get a one-dimensional fluorescence time-series, as shown in the bottom left panel (black line). By patching onto this neuron, the spike train can also be directly observed (black bars). Previous work suggests that this fluorescence signal might be well characterized by convolving the spike train with an exponential, and adding noise [1]. This model is confirmed by convolving the true spike train with an exponential (gray line, bottom left panel), and then looking at the distribution of the residuals. The bottom right panel shows a histogram of the residuals (black line), and the best fit Gaussian distribution (gray line).

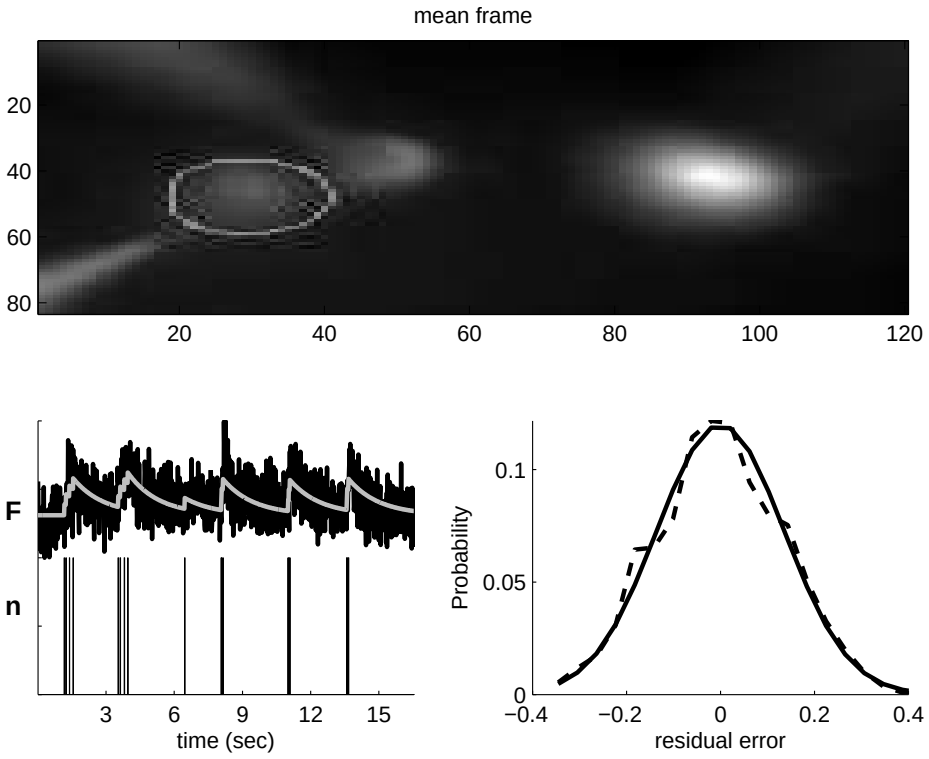

Figure 1: Typical in vitro data suggest that a reasonable first order model may be constructed by convolving the spike train with an exponential and adding Gaussian noise. Top panel: the average (over frames) of a typical field-of-view. Bottom left: spike train (black bars), convolved with an exponential (gray line), superimposed on the fluorescence trace (black line). Bottom right: a histogram of the residual error between the gray and black lines from the bottom left panel (dashed line), and the best fit Gaussian (solid line).

The above observations may be formalized as follows. Assume there is a one-dimensional fluorescence trace, $\boldsymbol{F}$ (throughout this text $\boldsymbol{X}$ indicates the vector $(X_1, \ldots, X_T)$, where T is the index of the final frame), from a neuron. At time t, the fluorescence measurement, F_t is a linear-Gaussian function of the intracellular calcium concentration at that time, $[\text{Ca}^{2+}]_t$:

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

The scale, α, absorbs all experimental variables impacting the scale of the signal, including number of sensors within the cell, photons per calcium ion, amplification of imaging system, etc. Similarly, the offset, β, absorbs baseline

calcium concentration of the cell, background fluorescence of the fluorophore, imaging system offset, etc. The standard deviation, σ, results from calcium fluctuations independent of spiking activity, fluorescence fluctuations independent of calcium, and imaging noise. The noise at each time, ε_t, is independently and identically distributed according to a standard normal distribution (ie, Gaussian with zero mean and unit variance), as indicated by the notation $\overset{iid}{\sim} \mathcal{N}(0,1)$.

Then, assuming that the intracellular calcium concentration, $[\text{Ca}^{2+}]_t$, jumps by A μM after each spike, and subsequently decays back down to C_b μM with time constant, τ, one can write:

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

where Δ is the time step size — which is the frame duration, or $1/$(frame rate) — and n_t indicates the number of times the neuron spiked in frame t. Note that because $[\text{Ca}^{2+}]_t$ and F_t are linearly related to one another, the fluorescence scale, α, and calcium scale, A, are not identifiable. In other words, either can be set to unity without loss of generality, as the other can absorb the scale entirely. Similarly, the fluorescence offset, β, and calcium baseline, C_b are not identifiable, so either can be set to zero without loss of generality. Finally, letting $\gamma = (1 - \Delta/\tau)$, Eq. (2) can be rewritten replacing $[\text{Ca}^{2+}]_t$ with its non-dimensionalized counterpart, C_t:

$$C_{t+1} = \gamma C_t + n_t. \tag{3}$$

Note that C_t does not refer to absolute intracellular concentration of calcium, but rather, a relative measure (but see [12] for a more general model). The gray line in the bottom left panel of Figure 1 corresponds to the putative $\boldsymbol{C}$ of the observed neuron.

To complete the "generative model" (ie, a model from which simulations can be generated), the distribution from which spikes are sampled must be defined. Perhaps the simplest first order description of spike trains is that at each time, spikes are sampled according to a Poisson distribution with some rate:

$$n_t \overset{iid}{\sim} \text{Poisson}(\lambda\Delta) \tag{4}$$

where $\lambda\Delta$ is the expected firing rate, and Δ is included to ensure that the expected firing rate is independent of the frame rate. Thus, Eqs. (1), (3), and (4) complete the generative model.

2.2 Goal

Given the above model, the goal is to find the maximum *a posteriori* (MAP) spike train, ie, the most likely spike train, $\widehat{\boldsymbol{n}}$, given the fluorescence measurements, $\boldsymbol{F}$:

$$\widehat{\boldsymbol{n}} = \underset{n_t \in \mathbb{N}_0 \forall t}{\text{argmax}}\, P[\boldsymbol{n}|\boldsymbol{F}], \tag{5}$$

where $P[\boldsymbol{n}|\boldsymbol{F}]$ is the posterior probability of a spike train, $\boldsymbol{n}$, given the fluorescent trace, $\boldsymbol{F}$, and n_t is constrained to be an integer, $\mathbb{N}_0 = \{0, 1, 2, \ldots\}$, because of the above assumed Poisson distribution. From Bayes' Rule, the posterior can be rewritten:

$$P[\boldsymbol{n}|\boldsymbol{F}] = \frac{P[\boldsymbol{n}, \boldsymbol{F}]}{P[\boldsymbol{F}]} = \frac{1}{P[\boldsymbol{F}]} P[\boldsymbol{F}|\boldsymbol{n}]P[\boldsymbol{n}], \tag{6}$$

where $P[\boldsymbol{F}]$ is the evidence of the data, $P[\boldsymbol{F}|\boldsymbol{n}]$ is the likelihood of observing a particular fluorescence trace $\boldsymbol{F}$, given the spike train $\boldsymbol{n}$, and $P[\boldsymbol{n}]$ is the prior probability of a spike train. Plugging the far right-hand-side of Eq. (6) into Eq. (5), yields:

$$\widehat{\boldsymbol{n}} = \underset{n_t \in \mathbb{N}_0 \forall t}{\text{argmax}} \frac{1}{P[\boldsymbol{F}]} P[\boldsymbol{F}|\boldsymbol{n}]P[\boldsymbol{n}] = \underset{n_t \in \mathbb{N}_0 \forall t}{\text{argmax}}\, P[\boldsymbol{F}|\boldsymbol{n}]P[\boldsymbol{n}], \tag{7}$$

where the second equality follows because $P[\boldsymbol{F}]$ merely scales the results, but does not change the relative quality of various spike trains. Fortunately, both $P[\boldsymbol{F}|\boldsymbol{n}]$ and $P[\boldsymbol{n}]$ are available from the above model:

$$P[\boldsymbol{F}|\boldsymbol{n}] = P[\boldsymbol{F}|\boldsymbol{C}] = \prod_{t=1}^{T} P[F_t|C_t], \tag{8a}$$

$$P[\boldsymbol{n}] = \prod_{t=1}^{T} P[n_t], \tag{8b}$$

where the first equality in Eq. (8a) follows because $\boldsymbol{C}$ is deterministic given $\boldsymbol{n}$, and the second equality follows from Eq. (1). Further, Eq. (8b) follows from the Poisson process assumption, Eq. (4). Both $P[F_t|C_t]$ and $P[n_t]$ can be written explicitly:

$$P[F_t|C_t] = \mathcal{N}(\alpha(C_t + \beta), \sigma^2), \tag{9a}$$

$$P[n_t] = \text{Poisson}(\lambda\Delta), \tag{9b}$$

where both equations follow from the above model. Now, plugging Eq. (9) back into (8), and plugging that result into Eq. (7), yields:

$$\widehat{\boldsymbol{n}} = \underset{n_t \in \mathbb{N}_0 \forall t}{\text{argmax}} \prod_{t=1}^{T} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2}\frac{(F_t - \alpha(C_t + \beta))^2}{\sigma^2}\right\} \frac{\exp\{-\lambda\Delta\}(\lambda\Delta)^{n_t}}{n_t!} \tag{10a}$$

$$= \underset{n_t \in \mathbb{N}_0 \forall t}{\text{argmax}} \sum_{t=1}^{T} \left(-\frac{1}{2\sigma^2}(F_t - \alpha(C_t + \beta))^2 + n_t \log \lambda\Delta - \log n_t! \right), \tag{10b}$$

where the second equality follows from taking the logarithm of the right-hand-side. Unfortunately, solving Eq. (10b) exactly is computationally intractable, as it requires a nonlinear search over an infinite number of possible spike trains. The search space could be restricted by imposing an upper bound, k, on the number of spikes within a frame. However, in that case, the computational complexity scales *exponentially* with the number of image frames — ie, the number of computations required would scale with k^T — which for pragmatic reasons is intractable.

2.3 Inferring the most likely spike train, given a fluorescence trace

The goal here is to develop an algorithm to efficiently approximate $\widehat{\boldsymbol{n}}$, the most likely spike train, given the fluorescence trace. Because of the computational intractability described above, Eq. (10) is approximated by modifying Eq. (4), replacing the Poisson distribution with an exponential distribution. Modifying Eq. (10) to reflect this approximation yields:

$$\widehat{\boldsymbol{n}} \approx \underset{n_t > 0 \forall t}{\text{argmax}} \prod_{t=1}^{T} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2}\frac{(F_t - \alpha(C_t + \beta))^2}{\sigma^2}\right\} (\lambda\Delta) \exp\{-\lambda\Delta n_t\} \tag{11a}$$

$$= \underset{n_t > 0 \forall t}{\text{argmax}} \sum_{t=1}^{T} -\frac{1}{2\sigma^2}(F_t - \alpha(C_t + \beta))^2 - n_t\lambda\Delta \tag{11b}$$

where the constraint on n_t has been relaxed from $n_t \in \mathbb{N}_0$ to $n_t \geq 0$ (since the exponential distribution can yield any non-negative number). The advantage of this approximation is that the optimization problem becomes log-concave in $\boldsymbol{C}$, meaning that any gradient ascent method guarantees achieving the global maximum (because there are no local maxima, other than the single global maximum). To see that Eq. (40) is log-concave in $\boldsymbol{C}$, rearrange Eq. (40) to obtain, $n_t = C_t - \gamma C_{t-1}$, so Eq. (40) can be rewritten:

$$\boldsymbol{C} = \underset{C_t - \gamma C_{t-1} > 0 \forall t}{\text{argmax}} \sum_{t=1}^{T} -\frac{1}{2\sigma^2}(F_t - \alpha(C_t + \beta))^2 - (C_t - \gamma C_{t-1})\lambda\Delta \tag{12}$$

which is a sum of terms that are log-concave in C_t, so the whole right-hand-side is log-concave. Unfortunately, the integer constraint has been lost, ie, the answer could include "partial" spikes. This disadvantage can be remedied by thresholding (ie, setting $n_t = 1$ for all n_t greater than some threshold, and the rest setting to zero), or by considering the magnitude of a partial spike at time t as the probability of a spike occurring during frame t. Note that replacing a Poisson with an exponential is a common approximation technique in the machine learning literature [21, 20], as the exponential distribution is the closest convex relaxation to its non-convex counterpart, the Poisson distribution. More specifically, the probability mass function of a Poisson distributed random variable with low rate is very similar to the probability density function of a random variable with an exponential distribution. While this convex relaxation makes the problem tractable, the "sharp" threshold imposed by the non-negativity constraint prohibits the use of standard gradient ascent techniques. This may be rectified by dropping the sharp threshold, and adding a barrier term,

which must approach $-\infty$ as n_t approaches zero (this approach is often called an "interior-point" method). Iteratively reducing the weight of the barrier term guarantees convergence to the correct solution. Thus, the goal is to efficiently solve:

$$\widehat{\boldsymbol{C}}_z = \underset{C_t - \gamma C_{t-1} \forall t}{\operatorname{argmax}} \sum_{t=1}^{T} \left(-\frac{1}{2\sigma^2}(F_t - \alpha(C_t + \beta))^2 - (C_t - \gamma C_{t-1})\lambda\Delta + z\log(C_t - \gamma C_{t-1}) \right). \tag{13}$$

where $\log(\cdot)$ is the "barrier term", and z is the weight of the barrier term. Iteratively solving for $\widehat{\boldsymbol{C}}_z$ for z going from one down to nearly zero, guarantees convergence to $\widehat{\boldsymbol{C}}$ [21]. The concavity of Eq. (13) facilitates utilizing any number of techniques guaranteed to find the global maximum. Because the argument of Eq. (13) is twice analytically differentiable, one can use the Newton-Raphson technique, which typically requires fewer steps than the gradient by itself [21]; and the special tridiagonal structure of the Hessian (as described below) enables each Newton-Raphson step to be very efficient. To proceed, Eq. (13) can be rewritten in matrix notation. Note that $\boldsymbol{MC} = \boldsymbol{n}$:

$$\boldsymbol{MC} = \begin{bmatrix} 1 & 0 & 0 & \cdots & \cdots \\ -\gamma & 1 & 0 & \cdots & \cdots \\ 0 & -\gamma & 1 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & -\gamma & 1 \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \\ \vdots \\ \vdots \\ C_T \end{bmatrix} = \begin{bmatrix} n_1 \\ n_2 \\ \vdots \\ \vdots \\ n_T \end{bmatrix} = \boldsymbol{n} \tag{14}$$

where $\boldsymbol{M} \in \mathbb{R}^{T \times T}$ is a bidiagonal matrix. Then, letting $\mathbf{1}$ be a T dimensional column vector and $\boldsymbol{\lambda} = \lambda\Delta\mathbf{1}^\mathsf{T}$, yields:

$$\widehat{\boldsymbol{C}}_z = \underset{\boldsymbol{MC} \geq \mathbf{0}}{\operatorname{argmax}} -\frac{1}{2\sigma^2} \left\| \boldsymbol{F} - \alpha(\boldsymbol{C} + \beta) \right\|^2 - (\boldsymbol{MC})^\mathsf{T}\boldsymbol{\lambda} + z\log(\boldsymbol{MC})^\mathsf{T}\mathbf{1}, \tag{15}$$

where $\boldsymbol{MC} \geq \mathbf{0}$ indicates that every element of $\boldsymbol{MC}$ is greater than or equal to zero, $^\mathsf{T}$ indicates the transpose operation, and $\log(\cdot)$ indicates an element-wise logarithm. When using Newton-Raphson to ascend a surface, one iteratively computes both the gradient (first derivative) and Hessian (second derivative) of the argument to be maximized, with respect to the variables of interest ($\boldsymbol{C}_z$ here). Then, the estimate is updated using $\boldsymbol{C}_z \leftarrow \boldsymbol{C}_z + s\boldsymbol{d}$, where s is the step size and $\boldsymbol{d}$ is the step direction (typically obtained by solving $\boldsymbol{Hd} = \boldsymbol{g}$). The gradient, $\boldsymbol{g}$, and Hessian, $\boldsymbol{H}$, for this model, with respect to $\boldsymbol{C}_z$, are given by:

$$\boldsymbol{g} = -\frac{\alpha}{\sigma^2}(\boldsymbol{F} - \alpha(\widehat{\boldsymbol{C}}_z^\mathsf{T} + \beta)) + \boldsymbol{M}^\mathsf{T}\boldsymbol{\lambda} - z\boldsymbol{M}^\mathsf{T}(\boldsymbol{M}\widehat{\boldsymbol{C}}_z)^{-1} \tag{16a}$$

$$\boldsymbol{H} = \frac{\alpha^2}{\sigma^2}\boldsymbol{I} + z\boldsymbol{M}^\mathsf{T}(\boldsymbol{M}\widehat{\boldsymbol{C}}_z)^{-2}\boldsymbol{M} \tag{16b}$$

where the exponents indicate element-wise operations. The step size, s, is found using "backtracking linesearches", which finds the maximal s that increases the posterior and is between zero and one.

Typically, implementing Newton-Raphson requires inverting the Hessian, ie, solving $\boldsymbol{d} = \boldsymbol{H}^{-1}\boldsymbol{g}$, a computation that scales *cubically* with T (requires on the order of T^3 operations). Already, this would be a drastic improvement over the most efficient algorithm assuming Poisson spikes, which would require k^T operations (where k is the maximum number of spikes per frame). Here, because $\boldsymbol{M}$ is bidiagonal, the Hessian is tridiagonal, so the solution may be found in about T operations, via standard banded Gaussian elimination techniques (which can be implemented efficiently in Matlab using $\boldsymbol{H}\backslash\boldsymbol{g}$, assuming $\boldsymbol{H}$ is represented as a sparse matrix). In other words, the above approximation and inference algorithm reduces computations from *exponential* time to *linear* time. Appendix A contains pseudocode for this algorithm, including learning the parameters, as described below.

2.4 Learning the parameters

While in the above, it was assumed that the parameters governing the model, $\boldsymbol{\theta} = \{\alpha, \beta, \sigma, \gamma, \lambda\}$, were known, in practice, they are typically unknown. An algorithm to estimate the most likely parameters, $\widehat{\boldsymbol{n}}$, could proceed as follows: (i) initialize some estimate of the parameters, $\widehat{\boldsymbol{\theta}}$, then (ii) recursively compute $\widehat{\boldsymbol{n}}$ using those parameters, and update $\widehat{\boldsymbol{\theta}}$ given the new $\widehat{\boldsymbol{n}}$, and (iii) stop recursing when some convergence criteria is met. Below, details are provided for each step.

2.4.1 Initializing the parameters

Because the above model is linear, the scale of $\boldsymbol{F}$ relative to $\boldsymbol{n}$ is arbitrary. Therefore, any change in the magnitude of α can be offset by a relative scaling of n_t. To ensure that the iterations do not get stuck in an infinite recursion of increasing α and scaling n_t down, we simply fix $\alpha = 1$. The offset, however, is not arbitrary. Because spiking is assumed to be sparse, $\boldsymbol{F}$ tends to be around baseline, so β is initialized to be the median of $\boldsymbol{F}$, and σ is initialized as median absolute deviation of $\boldsymbol{F}$, ie, σ =median$_t$(|F_t−median($\boldsymbol{F}$)|)/K, where K is the correction factor when using median absolute deviation as a robust estimator of the standard deviation. Because the posterior is relatively flat along the γ dimension, ie, large changes in γ result in relatively small changes in the posterior, estimating γ is difficult. Further, previous work has shown that results are somewhat robust to minor variations in time constant [22]; therefore γ is initialized at $1 - \Delta/(1\text{sec})$, which is fairly typical [23]. Finally, λ is initialized at 1 Hz, which is between typical baseline and evoked spike rate, for these data.

2.4.2 Estimating the parameters given $\widehat{\mathbf{n}}$

Ideally, one could integrate out the hidden variables, to find the most likely parameters:

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \iint P[\boldsymbol{F}, \boldsymbol{C}, \boldsymbol{n}|\boldsymbol{\theta}] d\boldsymbol{C} d\boldsymbol{n} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \int P[\boldsymbol{F}|\boldsymbol{C};\boldsymbol{\theta}] d\boldsymbol{C} \int P[\boldsymbol{n}|\boldsymbol{\theta}] d\boldsymbol{n}. \tag{17}$$

However, evaluating those integrals is not currently tractable (to our knowledge). Therefore, Eq. (17) is approximated by simply maximizing the parameters given the MAP estimate of the hidden variables:

$$\widehat{\boldsymbol{\theta}} \approx \underset{\boldsymbol{\theta}}{\operatorname{argmax}} P[\boldsymbol{F}, \widehat{\boldsymbol{C}}, \widehat{\boldsymbol{n}}|\boldsymbol{\theta}] = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} P[\boldsymbol{F}|\widehat{\boldsymbol{C}};\boldsymbol{\theta}] P[\widehat{\boldsymbol{n}}|\boldsymbol{\theta}] = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \log P[\boldsymbol{F}|\widehat{\boldsymbol{C}};\boldsymbol{\theta}] + \log P[\widehat{\boldsymbol{n}}|\boldsymbol{\theta}] \tag{18}$$

where $\widehat{\boldsymbol{C}}$ and $\widehat{\boldsymbol{n}}$ are determined using the above described inference algorithm. The approximation in Eq. (18) is good whenever the likelihood is very peaky, meaning that most of the mass is around the MAP sequence.[1] The argument from the right-hand-side of Eq. (18) may be expanded:

$$\log P[\boldsymbol{F}|\widehat{\boldsymbol{C}};\boldsymbol{\theta}] + \log P[\widehat{\boldsymbol{n}}|\boldsymbol{\theta}] = \sum_{t=1}^{T} \log P[F_t|\widehat{C}_t;\beta,\sigma] + \log P[\widehat{n}_t|\lambda]. \tag{19}$$

The maximum likelihood estimate (MLE) for $\widehat{\beta}$ and $\widehat{\sigma}$, is therefore given by:

$$\{\widehat{\beta},\widehat{\sigma}\} = \underset{\beta,\sigma>0}{\operatorname{argmax}} \sum_{t=1}^{T} \log P[F_t|\widehat{C}_t;\beta,\sigma] = \underset{\beta,\sigma>0}{\operatorname{argmax}} -\frac{1}{2}(2\pi\sigma^2) - \frac{1}{2}\left(\frac{F_t - (C_t+\beta)}{\sigma}\right)^2. \tag{20}$$

Maximizing Eq. (20) with respect to β yields

$$\widehat{\beta} = \underset{\beta>0}{\operatorname{argmax}} \sum_{t=1}^{T} -(F_t - (C_t+\beta))^2. \tag{21}$$

Computing the gradient with respect to β, setting the answer to zero, and solving for $\widehat{\beta}$, yields $\widehat{\beta} = \frac{1}{T}\sum_t (F_t - C_t)$. Similarly, computing the gradient of Eq. (20) with respect to σ, setting it to zero, and solving for $\widehat{\sigma}$ yields

$$\widehat{\sigma} = \sqrt{\frac{1}{T}\sum_t (F_t - C_t - \widehat{\beta})^2}. \tag{22}$$

Finally, the MLE of $\widehat{\lambda}$ is given by solving

$$\widehat{\lambda} = \underset{\lambda>0}{\operatorname{argmax}} \sum_t \log(\lambda\Delta) - n_t\lambda\Delta, \tag{23}$$

which, again, computing the gradient with respect to λ, setting it to zero, and solving for $\widehat{\lambda}$, yields $\widehat{\lambda} = \frac{1}{T\Delta}\sum_t n_t$.

[1] Eq. (18) may be considered a first-order Laplace approximation

2.4.3 Convergence criteria

Iterations stop whenever (i) iteration number exceeds some upper bound, or (ii) relative change in likelihood does not exceed some lower bound. In practice, parameters tend to converge after several iterations, given the above initializations.

2.5 Spatial filtering

In the above, the raw movie of fluorescence measurements collected by the experimenter had undergone two stages of preprocessing before filtering. First, the movie was segmented, to determine regions-of-interest (ROIs), yielding a vector, $\vec{F}_t = (F_{1,t}, \ldots, F_{N_p,t})$, which corresponded to the fluorescence intensity at time t for each of the N_p pixels in the ROI. Second, at each time t, that vector was projected into a scalar, yielding F_t, the assumed input to the filter. In this section, the optimal projection is determined by considering a more general model:

$$F_{x,t} = \alpha_x(C_t + \beta) + \sigma\varepsilon_{x,t}, \qquad \varepsilon_{x,t} \overset{iid}{\sim} \mathcal{N}(0,1) \tag{24}$$

where α_x scales each pixel, from which some number of photons are contributed due to calcium fluctuations, C_t, and others due to baseline fluorescence, β. Further, the noise is assumed to be both spatially and temporally white, with standard deviation, σ, in each pixel (this assumption can easily be relaxed, by making the covariance matrix diagonal, for example). Performing inference in this more general model proceeds in a nearly identical manner as before. In particular, the maximization, gradient, and Hessian become:

$$\widehat{\boldsymbol{C}}_z = \underset{\boldsymbol{MC}\geq\boldsymbol{0}}{\operatorname{argmax}} -\frac{1}{2\sigma^2}\left\|\vec{\boldsymbol{F}} - \vec{\alpha}(\boldsymbol{C}^\mathsf{T} + \beta\boldsymbol{1}^\mathsf{T})\right\|^2 - (\boldsymbol{MC})^\mathsf{T}\boldsymbol{\lambda} + z\log(\boldsymbol{MC})^\mathsf{T}\boldsymbol{1}, \tag{25}$$

$$\boldsymbol{g} = \frac{\vec{\alpha}}{\sigma^2}(\vec{\boldsymbol{F}} - \vec{\alpha}(\widehat{\boldsymbol{C}}_z^\mathsf{T} + \beta)) - \boldsymbol{M}^\mathsf{T}\boldsymbol{\lambda} + z\boldsymbol{M}^\mathsf{T}(\boldsymbol{M}\widehat{\boldsymbol{C}}_z)^{-1} \tag{26}$$

$$\boldsymbol{H} = -\frac{\vec{\alpha}^\mathsf{T}\vec{\alpha}}{\sigma^2}\boldsymbol{I} - z\boldsymbol{M}^\mathsf{T}(\boldsymbol{M}\widehat{\boldsymbol{C}}_z)^{-2}\boldsymbol{M} \tag{27}$$

where $\vec{\boldsymbol{F}}$ is an $N_p \times T$ element matrix, $\vec{\alpha}$ is column vectors of length N_p, $\boldsymbol{1}$ is a column vector of appropriate length, and $\boldsymbol{I}$ is an $N_p \times N_p$ identity matrix. Typically, the parameters $\vec{\alpha}$ and β are unknown, and therefore must be estimated from the data. The maximum likelihood estimator for each α_x is given by:

$$\widehat{\alpha}_x = \underset{\alpha_x}{\operatorname{argmax}}\, P[\boldsymbol{F}_x|C_t] = \underset{\alpha_x}{\operatorname{argmax}} \sum_t \log P[F_{x,t}|C_t] \tag{28a}$$

$$= \underset{\alpha_x}{\operatorname{argmax}} \sum_t -\frac{1}{2}(2\pi\sigma^2) - \frac{1}{2}\left(\frac{F_{x,t} - \alpha_x(C_t + \widehat{\beta})}{\sigma}\right)^2 = \underset{\alpha_x}{\operatorname{argmax}} \sum_t -(F_t - \alpha_x(C_t - \widehat{\beta}))^2, \tag{28b}$$

Upon computing the gradient with respect to α_x and setting to zero, one obtains $\widehat{\alpha}_x = \frac{1}{T}\sum_t F_{x,t}/(C_t - \widehat{\beta})$. Because the denominator is *not* a function of x, and the scale of C_t relative to F_t is arbitrary, this is equivalent to $\widehat{\alpha}_x = \frac{1}{T}\sum_t F_{x,t}$, ie, the optimal spatial filter is obtained from the mean intensity of each pixel. Computing the gradient of Eq. (28b) with respect to $\widehat{\beta}$, setting the result to zero, and solving for $\widehat{\beta}$, yields

$$\widehat{\beta} = \frac{1}{T}\sum_t \sum_x \frac{F_{x,t}}{\alpha_{x,t}} - C_t. \tag{29}$$

2.6 Overlapping spatial filters

In the above, the image was segmented before filtering, such that only a single neuron was within each ROI. However, segmentation is itself a difficult problem [24]. Therefore, using a crude segmentation technique, that might not actually produce ROIs with only a single cell, and then building spatial filters for each neuron in the ROI, might both be more efficient and improve SNR. This requires a minor modification to the model. Specifically, letting the superscript i

index the N_c neurons in this ROI, yields:

$$\vec{F}_t = \sum_{i=1}^{N_c} \vec{\alpha}^i (C_t^i + \beta^i) + \sigma \vec{\varepsilon}_t, \qquad \vec{\varepsilon}_t \overset{iid}{\sim} \mathcal{N}(\mathbf{0}, \boldsymbol{I}) \tag{30}$$

$$C_t^i = \gamma^i C_{t-1}^i + n_t^i, \qquad n_t^i \overset{iid}{\sim} \text{Poisson}(n_t^i; \lambda_i \Delta) \tag{31}$$

where each neuron is implicitly assumed to be independent, and each pixel is independent and identically distributed with standard deviation σ. To perform inference in this more general model, let:

$$\boldsymbol{n} = [n_1^1, n_1^2, \ldots, n_1^{N_c}, n_2^1, \ldots, n_T^{N_c}]^\mathsf{T} \tag{32}$$

$$\boldsymbol{C} = [C_1^1, C_1^2, \ldots, C_1^{N_c}, n_2^1, \ldots, C_T^{N_c}]^\mathsf{T} \tag{33}$$

$$\boldsymbol{M} = \begin{bmatrix} 1 & 1 & 0 & 0 & \cdots & \cdots & & & \\ -\gamma^1 & 1 & -\gamma_2 & 1 & \ldots & -\gamma_{N_c} & 1 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & & & & \\ 0 & 0 & 0 \ldots & -\gamma^{N_c - 1} & 1 & -\gamma^{N_c} & 1 & & \end{bmatrix} \tag{34}$$

and proceed as above, making minor algorithmic adjustments to deal with dimensionality issues. Note that $\boldsymbol{M}$ is no longer tridiagonal, but rather, block tridiagonal. Importantly, the Thomas algorithm, which is a simplified form of Gaussian elimination, finds the solution to linear equations with block tridiagonal matrices in linear time, so the efficiency gained from utilizing the tridiagonal structure above is maintained for this block tridiagonal structure [25].

If the parameters are unknown, they must be estimated. Define $\boldsymbol{\alpha}_x = [\alpha_x^1, \ldots, \alpha_x^{N_c}]^\mathsf{T}$ and $\boldsymbol{\beta} = [\beta^1, \ldots, \beta^{N_c}]^\mathsf{T}$. To initialize, let $\boldsymbol{\beta} = \mathbf{0}$. To initialize $\vec{\boldsymbol{\alpha}} = (\vec{\alpha}_1, \ldots, \vec{\alpha}_{N_c})$, the goal is to be able to represent the matrix, $\vec{\boldsymbol{F}}$, as the sum of only N_c time-varying components, also known as a *low-rank* approximation. Singular value decomposition is a tool known to find the low-rank approximation to a matrix, with the smallest mean-square-error of all possible low-rank approximations [26]. Because the "singular values" are equivalent to the "principal components" of the covariance of the movie (obtained from doing principal component analysis, aka, PCA), a natural initial estimate for the N_c $\vec{\alpha}$ vectors is the N_c first principal components. While other methods to initialize the spatial filters (such as "independent component analysis") could also work, because fast algorithms for computing the first few principal components are readily available [27], PCA was found to be both effective and efficient. Given these initializations, estimating $\vec{\alpha}$ and $\boldsymbol{\beta}$ follows very similarly as in Eqs. (28b) and (29):

$$\widehat{\boldsymbol{\alpha}}_x = \underset{\boldsymbol{\alpha}_x}{\text{argmin}} \sum_{t=1}^{T} \left(F_{x,t} \sum_{i=1}^{N_c} \alpha_x^i (C_t^i + \beta^i) \right)^2 \tag{35}$$

$$\widehat{\beta}^i = \frac{1}{T} \sum_t (\vec{F}_t / \vec{\alpha}^i - C_t^i) \tag{36}$$

where Eq. (35) is solved efficiently in Matlab using $\boldsymbol{\alpha}_x = (\boldsymbol{C} + \widetilde{\boldsymbol{\beta}}) \backslash \boldsymbol{F}_x$. In practice, iterating these two steps converged after several iterations, assuming enough spikes were present in the two neurons, and they were sufficiently uncorrelated. Note that updating each α_x is more efficient then solving for the entire vector, $\vec{\alpha}$, due to the quadratic structure of the problem.

2.7 Experimental Methods

2.7.1 Slice Preparation and Imaging

All animal handling and experimentation was done according to the National Institutes of Health and local Institutional Animal Care and Use Committee guidelines. Somatosensory thalamocortical or coronal slices 350-400 μm thick were prepared from C57BL/6 mice at age P14 as described [28]. Neurons were filled with 50 μM Oregon Green Bapta 1 hexapotassium salt (OGB-1; Invitrogen, Carlsbad, CA) through the recording pipette or bulk loaded with Fura-2 AM (Invitrogen, Carlsbad, CA). Pipette solution contained 130 K-methylsulfate, 2 $MgCl_2$, 0.6 EGTA, 10 HEPES, 4 ATP-Mg, and 0.3 GTP-Tris, pH 7.2 (295 mOsm). After cells were fully loaded with dye, imaging was done by using a modified BX50-WI upright microscope (Olympus, Melville, NY). Image acquisition was performed with the

C9100-12 CCD camera from Hamamatsu Photonics (Shizuoka, Japan) with arclamp illumination with excitation and emission bandpass filters at 480-500 nm and 510-550 nm, respectively (Chroma, Rockingham, VT) for Oregon Green. Imaging of Fura-2 loaded slices was performed with a confocal spinning disk (Solamere Technology Group, Salt Lake City, UT) and an Orca CCD camera from Hamamatsu Photonics (Shizuoka, Japan). Images were saved and analyzed using custom software written in Matlab (Mathworks, Natick, MA).

2.7.2 Electrophysiology

All recordings were made using the Multiclamp 700B amplifier (Molecular Devices, Sunnyvale, CA), digitized with National Instruments 6259 multichannel cards and recorded using custom software written using the LabView platform (National Instruments, Austin, TX) . Square pulses of sufficient amplitude to yield the desired number of action potentials were given as current commands to the amplifier using the LabView and National Instruments system.

2.7.3 Fluorescence preprocessing

Traces were extracted using custom Matlab scripts to (i) segment the mean image into ROIs, and then (ii) average all the pixels within the ROI. The Fura-2 fluorescence traces were inverted. As some slow drift was sometimes present in the traces, the lowest ten frequency components were set to zero (ten was chosen by eye), and the resulting fluorescence trace was then normalized to be between zero and one.

3 Results

3.1 Main Result

The main result of this paper is that the fast filter can approximate $\widehat{n}$ very efficiently, and that this approach yields more accurate spike train estimates than optimal linear deconvolution. Fig. 2 depicts a simulation showing this result. Clearly, the fast filter is outperforming the optimal linear deconvolution (also called a Wiener filter). The Wiener filter implicitly approximates the Poisson spike rate with a Gaussian spike rate (see Appendix B for details). While a Gaussian well approximates a Poisson distribution when rates are about 10 spikes per frame, this example is very far from that regime, and so the Gaussian approximation performs relatively poorly. Furthermore, the Gaussian approximation allows for the inferred spike train to include negative numbers, which is undesirable, as spike trains are non-negative entities. To counteract the negative values, the Wiener filter then infers large positive values, contributing to a "ringing" effect as seen in the bottom panel. The non-negative constraint imposed by the fast filter ensures that such ringing does not take place. Finally, by utilizing Gaussian elimination and interior-point methods, as described in the Methods section, the computational complexity of fast filter is the same as an efficient implementation of the Wiener filter.

Although in Figure 2 the model parameters were provided, in the general case, the parameters are unknown, and must therefore be estimated from the observations (as described in section 2.4). Importantly, estimating the parameters in this "unsupervised" manner obviates the need to conduct joint imaging and electrophyiological experiments to obtain "training" data. Figure 3 shows another simulated example; in this example, however, the parameters are estimated from the observed fluorescence trace alone. Again, it is clear that the fast filter far outperforms the Wiener filter.

Given the above two results, the fast filter was applied to real data. More specifically, by jointly recording electrophysiologically and imaging, the true spike times are known, and the accuracy of the two filters can be compared. Figure 4 shows a result typical of the 12 joint electrophysiological and imaging experiments conducted. Note that the first few "events" are actually pairs of spikes, which is reflected in the inferred spike trains. This suggests that although the scale is arbitrary, the fast filter can correctly ascertain the number of spikes within spike events.

Figure 5 further evaluates this claim. The cell was forced to spike in "doublets" for each spiking event, meaning that the algorithm never saw a single spike event. Naïvely, this would suggest that a purely linear approach would struggle to resolve spike fidelity in this high frequency spiking regime. However, the fast filter correctly identifies nearly all spiking events as "doublets". On the contrary, there is no obvious way to count the number of spikes within each event upon using the Wiener filter.

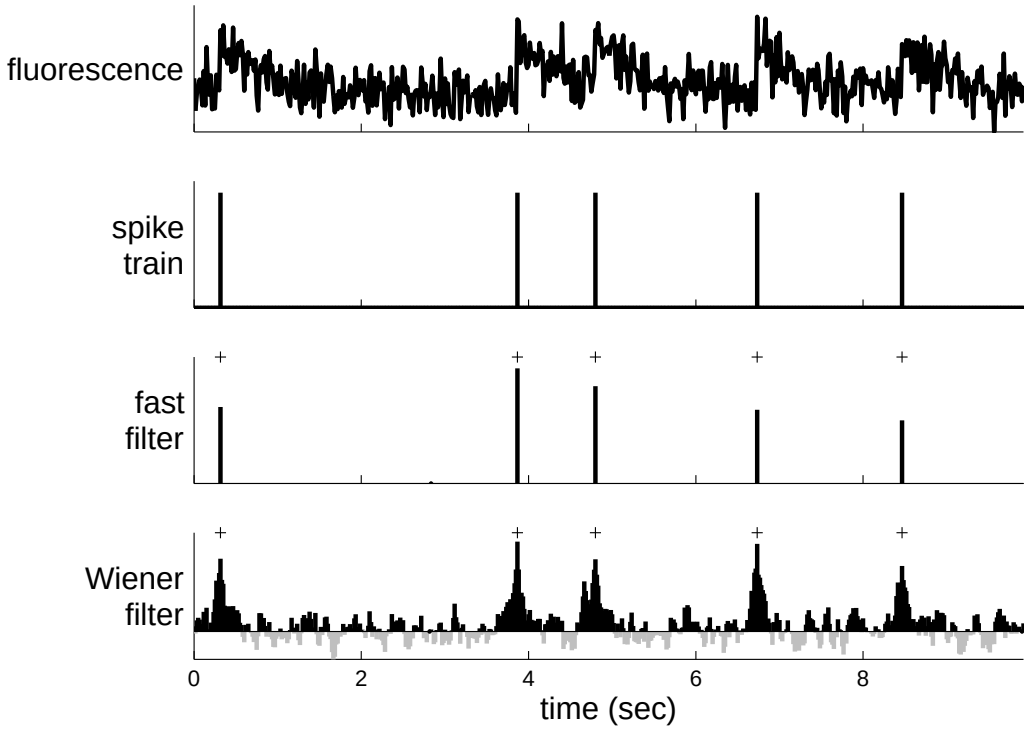

Figure 2: The fast filter significantly outperforms optimal linear deconvolution (aka, the Wiener filter) on typical simulated data. Top panel: fluorescence trace. Second panel: spike train. Third panel: fast filter inference. Bottom panel: Wiener filter inference. Note that the gray bars in the bottom panel indicate *negative* spikes, which, of course, are meaningless. Black '+'s in bottom two panels indicate true spike times. Simulation details: $T \approx 3000$ time steps, $\Delta = 5$ msec, $\alpha = 1$, $\beta = 0$, $\sigma = 0.3$, $\tau = 1$ sec, $\lambda = 1$ Hz.

3.2 Online analysis of spike trains using the fast filter

A central aim for this work was the development of an algorithm that infers spikes fast enough to use online while imaging a large population of neurons (eg, ≈ 100). Figure 6 shows a segment of the results of running the fast filter on 136 neurons, recorded simultaneously, as described in section 2.7. Note that the filtered fluorescence signals show fluctuations in spiking much more clearly than the unfiltered fluorescence trace. These spike trains were inferred in super-real-time, meaning that one could infer spike trains for the past experiment while conducting the subsequent experiment. More specifically, a movie with 5,000 frames of 100 neurons can be analyzed in about ten seconds on a standard desktop computer. Thus, if that movie was recorded at 50 Hz, while collecting the data required 100 seconds, inferring spikes only required ten seconds, a ten-fold improvement over real-time.

3.3 Extensions

Section 2.1 describes a simple principled first-order model relating the spike trains to the fluorescence trace. A number of the simplifying assumptions can be straightforwardly relaxed, as described below.

3.3.1 Replacing Gaussian observations with Poisson

In the above, observations were assumed to have a Gaussian distribution. The statistics of photon emission and counting, however, suggest that a Poisson distribution would be more natural [29], yielding:

$$F_t \overset{iid}{\sim} \text{Poisson}(\alpha C_t + \beta). \tag{37}$$

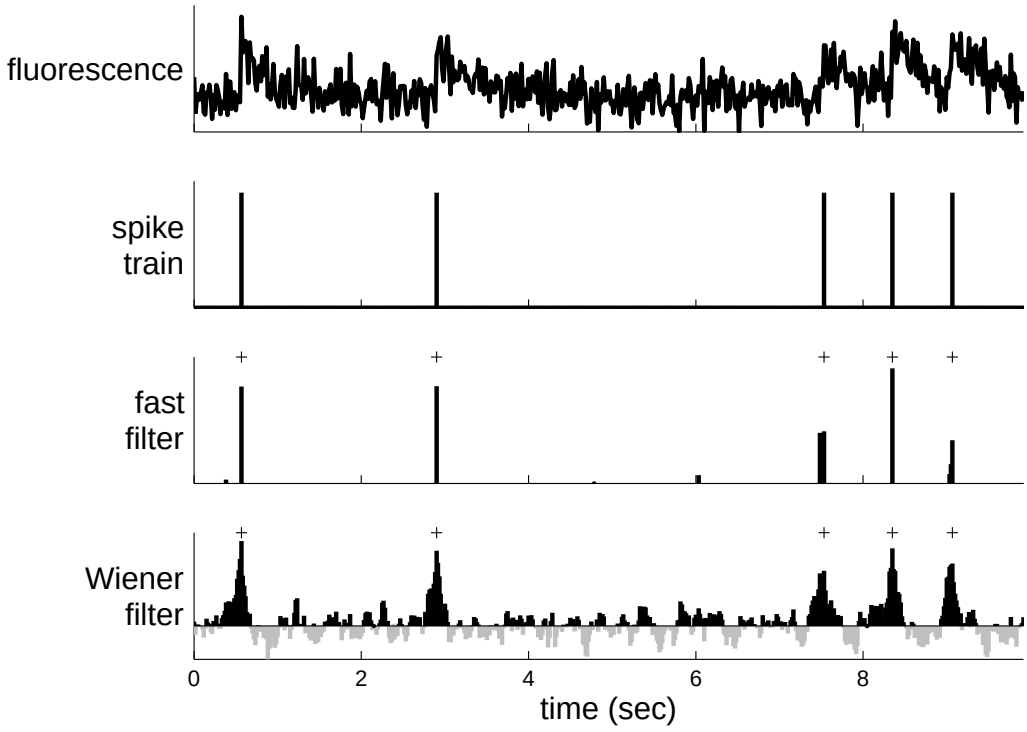

Figure 3: The fast filter significantly outperforms the Wiener filter, even when estimating the parameters only from the observed simulated data. Simulation details as in Figure 2.

One advantage to this model, over the Gaussian one, besides accuracy, is that the variance parameter no longer exists, which might make learning the parameters simpler. Importantly, the posterior is still concave in $\boldsymbol{C}$, so the same techniques can be used for this model (which requires analytically calculating the gradient and Hessian for the likelihood term implied by Eq. (37)). In practice, modifying the filter for this model extension did not seem to improve inference results in any simulations or data (not shown).

3.3.2 Allowing for a time-varying prior

In Eq. (4), the rate of spiking is a constant. Often, additional knowledge about the experiment, including external stimuli, or other neurons spiking, can provide strong time-varying prior information. A simple model modification can incorporate that feature:

$$n_t \overset{iid}{\sim} \text{Poisson}(\lambda_t \Delta), \tag{38}$$

where λ_t is now a function of time. Approximating this time-varying Poisson with a time-varying exponential (as in Eq. (10)) , and letting $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_T)\Delta$, yields an objective function identical to Eq. (15), so log-concavity is maintained, and the same techniques may be applied. However, as above, this model extension did not yield any improved filtering results (not shown).

3.3.3 Saturating fluorescence

Although all the above models assumed a *linear* relationship between F_t and C_t, the relationship between fluorescence and calcium is typically better approximated by the nonlinear Hill equation [23]. Modifying Eq. (1) to reflect this

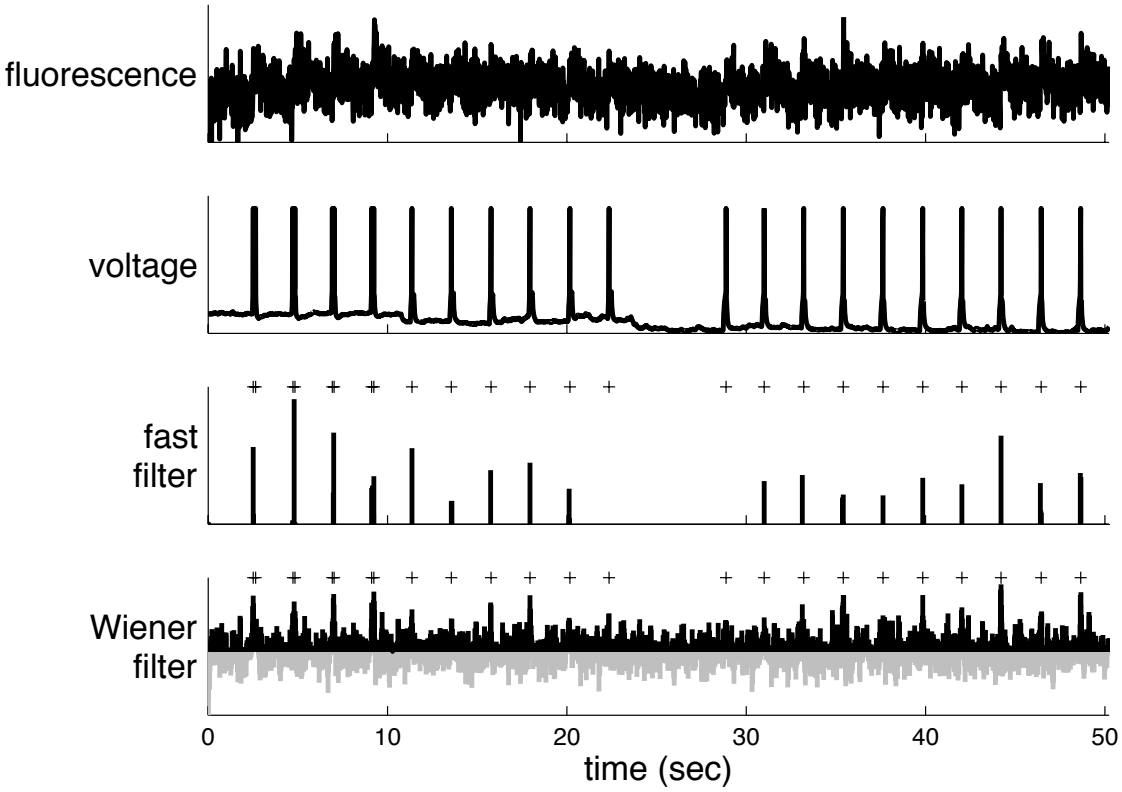

Figure 4: The fast filter significantly outperforms the Wiener filter on typical in vitro data, using Fura-2. Note that all the parameters for both filters were estimated from the fluorescence data alone (ie, not considering the voltage data at all).

change yields:

$$F_t = \alpha \frac{C_t}{C_t + k_d} + \beta + \sigma\varepsilon_t, \qquad \varepsilon_t \overset{iid}{\sim} \mathcal{N}(0,1). \tag{39}$$

Incorporating this nonlinearity breaks the log-concavity of the posterior, meaning that converging to the global maximum is no longer guaranteed. Assuming a good initialization can be found, however, if this model is more accurate, then ascending the gradient for this model might yield improved inference results. In practice, initializing with the inference from the fast filter assuming a linear model (eg, Eq. (30)) often resulted in nearly equally accurate inference, but inference assuming the above nonlinearity was far less robust than the inference assuming the linear model (not shown).

3.3.4 Using the fast filter to initialize the SMC filter

A previously proposed sequential Monte Carlo (SMC) method to infer spike trains can incorporate this nonlinearity, as well as the other model extensions discussed above [12] . However, this SMC filter is not nearly as efficient as the fast filter proposed here. Like the fast filter, the SMC filter estimates the model parameters in a completely unsupervised fashion, ie, from the fluorescence observations, using an expectation-maximization algorithm (which requires iterating between computing the expected value of the hidden variables — $\boldsymbol{C}$ and $\boldsymbol{n}$ — and updating the paramters). In [12], parameters for the SMC filter were initialized based on other data. While effective, this initialization was often far from the final estimates, and therefore, required a relatively large number of iterations (eg, 20–25) before converging. Thus, it seemed that the fast filter could be used to obtain an improvement to the initial parameter estimates, reducing the required number of iterations. Indeed, Figure 7 shows how the SMC filter outperforms the fast filter on in vitro data, and only required 3–5 iterations to converge on this data (which was typical). Note that the first few events of

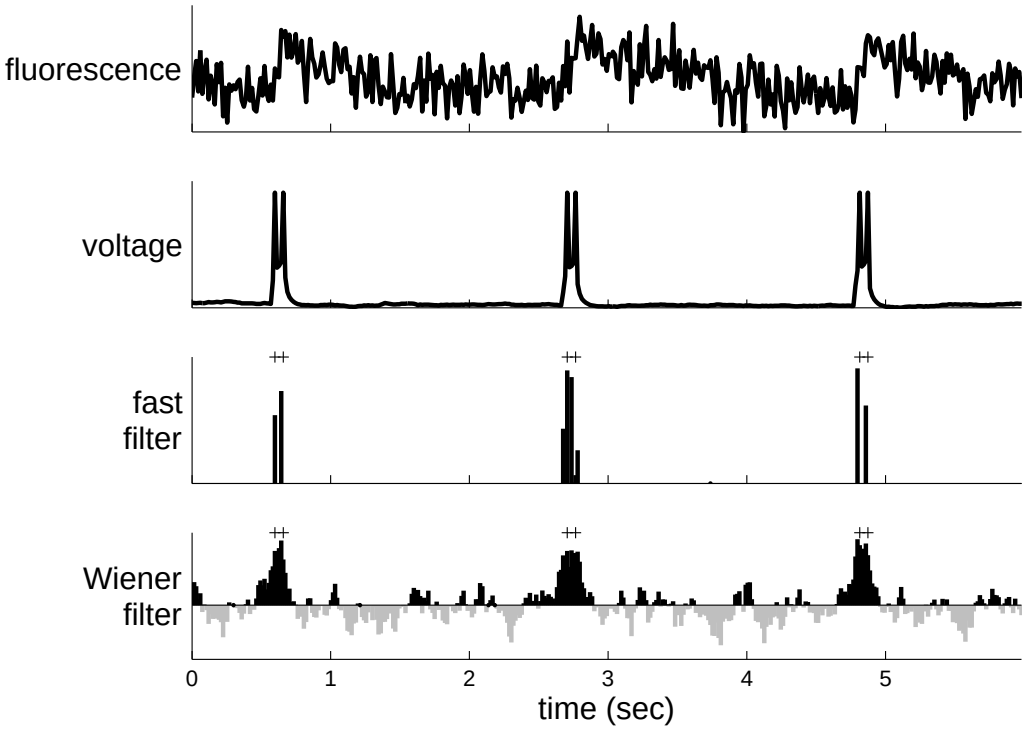

Figure 5: The fast filter can resolve a sequence of doublets, even in the absence of ever seeing an individual spike. It is difficult, if not impossible, to count the number of spikes given the Wiener filter output. Recording and fitting parameters as in Figure 4.

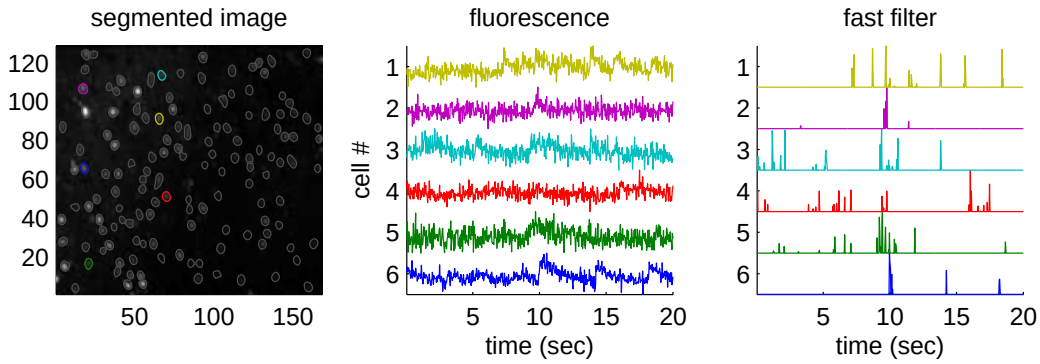

Figure 6: The fast filter infers spike trains from a large population of neurons imaged simultaneously in vitro, using OBG-1, in super-real-time. Specifically, inferring the spike trains from this 400 sec long movie requires only about 40 sec on a standard laptop computer. The inferred spike trains much more clearly convey neural activity than the raw fluorescence traces. Left panel: Mean segmented image field. Middle panel: example fluorescence traces. Right panel: fast filter output corresponding to each associated trace.

the spike train are individual spikes, resulting in relatively small fluorescence fluctuations, whereas the next events are actually spike doublets or triplets, causing a much larger fluorescence fluctuation. Only the SMC filter picks up the individual spikes in this trace, a result typical when the effective signal-to-noise ratio (SNR) is so poor. Thus, these two inference algorithms are complementary: the fast filter can be used for rapid, online inference, and for initializing

the SMC filter, which can then be used to further refine the spike train estimate. Importantly, although the SMC filter often outperforms the fast filter, the fast filter is more robust, meaning that it more often works "out-of-the-box".

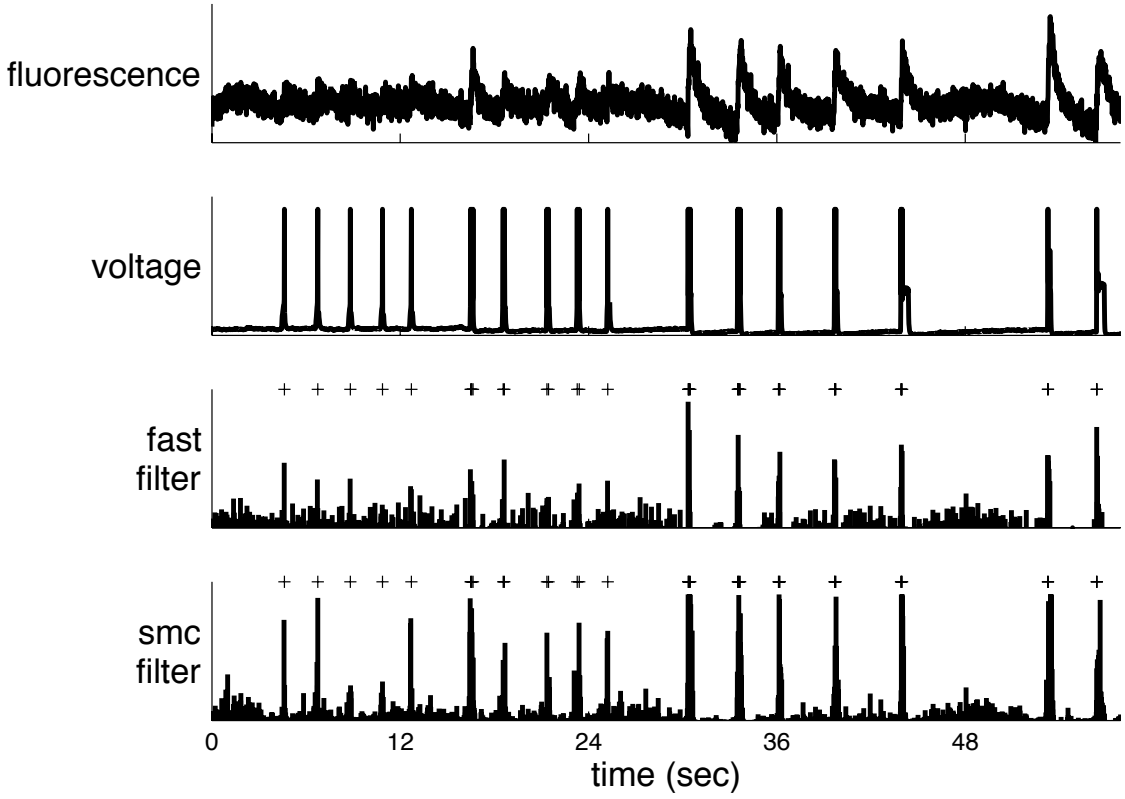

Figure 7: The fast filter effectively initializes the parameters for the SMC filter (which outperforms the fast filter), significantly reducing the number of expectation-maximization iterations to convergence, on typical in vitro data, using Fura-2. Note that the ordinate on the bottom panel corresponds to the inferred probability of a spike having occurred in each frame.

3.4 Spatial filter

In the above, the filters operated on one-dimensional fluorescence traces. Typically, although the data are time-series of images which are first segmented into regions-of-interest (ROI), and then (usually) averaged to obtain F_t. In theory, one could improve the effective SNR of the fluorescence trace by scaling each pixel relative to one another. In particular, pixels not containing any information about calcium fluctuations can be ignored, and pixels that are partially anti-correlated with one another could have weights with opposing signs.

Figure 8 demonstrates the potential utility of this approach. The top row shows different depictions of an ROI containing a single neuron. On the far left panel is the true spatial filter for this neuron. This particular spatial filter was chosen based on experience analyzing both in vitro and in vivo movies; often, it seems that the pixels immediately around the soma are anti-correlated with those in the soma. This effect is possibly due to the influx of calcium from extracellular space immediately around the soma. The simulated movie is relatively noisy, as indicated by the second panel, which depicts an exemplary image frame. The standard approach, given such a noisy movie, would be to first segment the movie to find an ROI corresponding to the soma of this cell, and then spatially average all the pixels found to be within this ROI. The third panel shows this "typical spatial filter". The fourth panel shows the mean frame. Clearly, this mean frame is very similar to the true spatial filter.

The bottom panels of Figure 8 depict the effect of using the true spatial filter, versus the typical one. The left side shows the fluorescence trace and its associated spike inference obtained from using the typical spatial filter. The right side shows the same when using the true spatial filter. Clearly, the true spatial filter results in a much cleaner

fluorescence trace and spike inference. When the true spatial filter is a single Gaussian, the typical spatial filter works about as well as the true one (not shown).

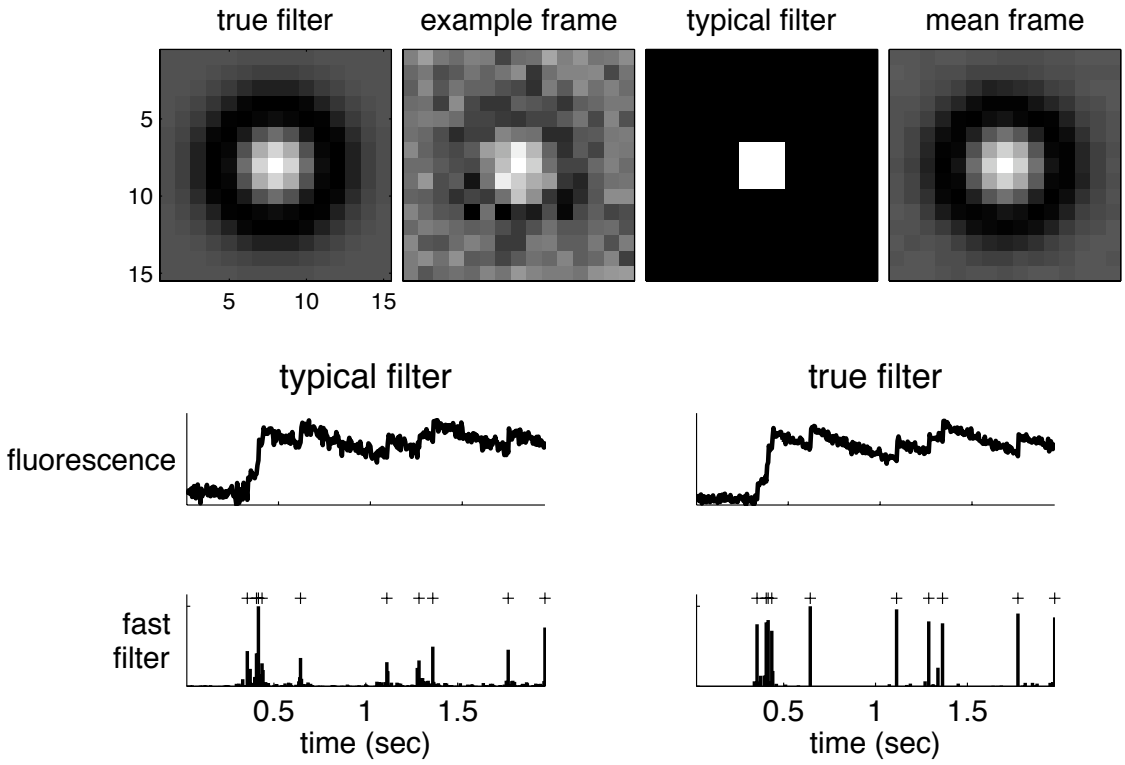

Figure 8: A simulation demonstrating that using a better spatial filter can significantly enhance the effective SNR. The true spatial filter was a sum of Gaussians: a positively weighted small variance Gaussian, and a negatively weighted large variance Gaussian (both with the same mean). Top row far left: true spatial filter. Top row second from left: example frame (frame number 100). Top row second from right: typical spatial filter. Top row far right: mean frame. Middle row left: fluorescence trace using typical spatial filter. Bottom row left: fast filter output using typical spatial filter. Middle row right: fluorescence trace using true spatial filter. Bottom right: fast filter output using true spatial filter. Simulation details: $\vec{\alpha} = \mathcal{N}(\mathbf{0}, 2\boldsymbol{I}) - 1.1\mathcal{N}(\mathbf{0}, 2.5\boldsymbol{I})$ where $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ indicates a Gaussian with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, $\boldsymbol{\beta} = 1$, $\tau = 0.85$ sec, $\lambda = 5$ Hz.

3.5 Overlapping spatial filters

The above shows that if a ROI contains only a single neuron, the effective SNR can be enhanced by spatially filtering. However, this analysis assumes that only a single neuron is in the ROI. Often, neural spatial filters are overlapping, or nearly overlapping, making the segmentation problem even more difficult. Therefore, it is desirable to have an ability to crudely segment, yielding only a few neurons in each ROI, and then spatially filter within each ROI to pick out the spike trains from each neuron. This may be achieved in a principled manner by generalizing the model as described in section 2.6. Figure 9 shows how this approach can separate the two signals, assuming that the spatial filters of the two neurons is known. Figure 10 shows that the spatial filters can be estimated using only the fluorescence movie, by using the approach described in section 2.6.

4 Discussion

This work describes an algorithm that approximates the *maximum a posteriori* (MAP) spike train, given a fluorescence movie. The approximation is required because finding the actual MAP estimate is not currently computationally

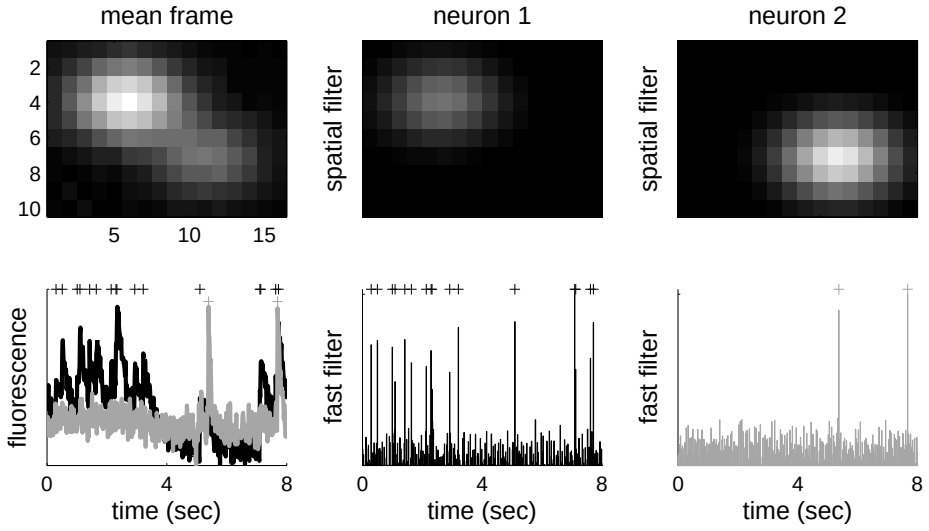

Figure 9: Simulation showing that even when two neurons' spatial filters are overlapping, one can separate the two spike trains by spatial filtering. Top left panel: mean frame from the movie. Bottom left: optimal one-dimensional fluorescence projections for the neuron 1 (black line) and neuron 2 (gray line), and their respective spike trains (black and gray plusses, respectively). Top middle panel: the true spatial filter for neuron 1. Bottom middle panel: inferred (black line) and true (black plusses) spike trains. Top right panel: the true spatial filter for neuron 2. Bottom right panel: inferred (gray line) and true (gray plusses) spike trains. Simulation details: $\vec{\alpha}^1 = \mathcal{N}([-1.8, 1.8]^\mathsf{T}, 2\boldsymbol{I})\, \vec{\alpha}^2 = \mathcal{N}([1.8, -1.8]^\mathsf{T}, 5\boldsymbol{I})$, $\boldsymbol{\beta} = [1, 1]^\mathsf{T}$, $\tau = [0.5, 0.5]^\mathsf{T}$ sec, $\lambda = [1.5, 1.5]^\mathsf{T}$ Hz.

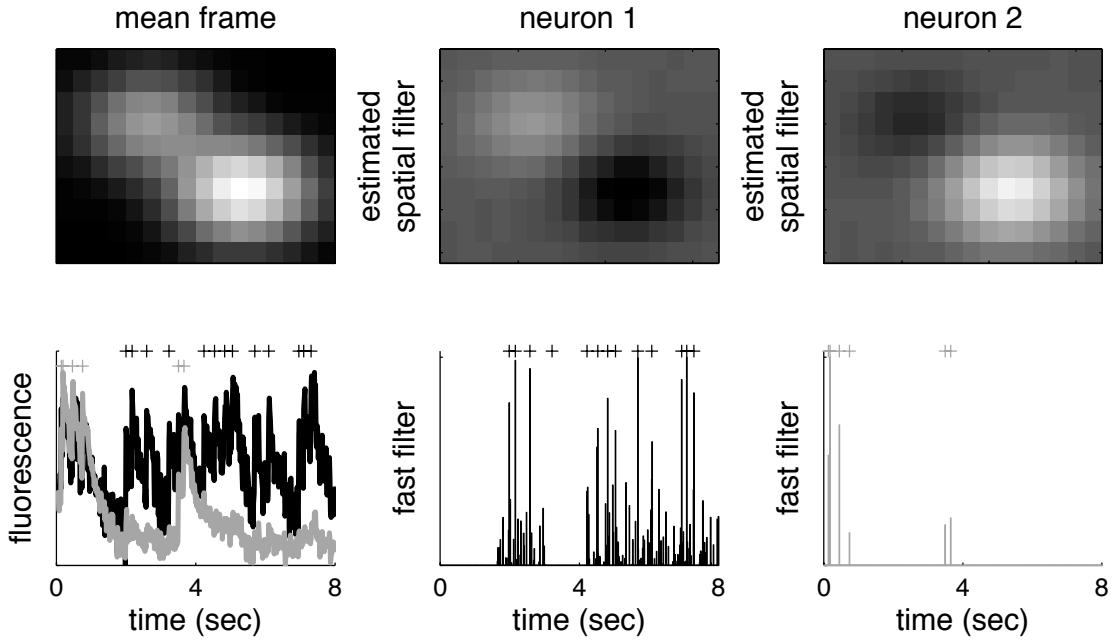

Figure 10: Simulation showing that even when two neuron's spatial filters are largely overlapping, linearly scaled estimates of the spatial filter of each neuron can be inferred, to separate the two signals. Simulation details as above. Note that the spatial fields are sufficiently overlapping to cause significant "bleed-through" between the two signals. In particular, this is clear from the rise in the black line in the first few seconds of the bottom left panel, which should be fluorescence due to neuron 1, but is in fact due to spiking from neuron 2.

tractable. Replacing the assumed Poisson distribution on spikes with an exponential distribution yields a log-concave optimization problem, which can be solved using standard gradient ascent techniques (such as Newton-Raphson). This exponential distribution has an advantage over a Gaussian distribution by restricting spikes to be positive, which improves inference quality (c.f. Figure 2), and is a better approximation to a Poisson distribution with low rate. Furthermore, by utilizing the special structure of the Hessian matrix (ie, it is tridiagonal), this approximate MAP spike train can be inferred fast enough on standard computers to use it for online analyses. Finally, all the parameters can be estimated from only the fluorescence observations, obviating the need for joint electrophysiology and imaging (c.f. Figure 3). This approach is robust, in that it works "out-of-the-box" on all the in vivo and in vitro data analyzed (c.f. Figure 4).

Ideally, one could compute the full joint posterior of entire spike trains, conditioned on the fluorescence data. This distribution is analytically intractable, due to the Poisson assumption on spike trains. A Bayesian approach could use Markov Chain Monte Carlo to recursively sample spikes until a whole sample spike train is obtained [30, 31]. Because a central aim here was computational expediency, a "greedy" approach is natural: ie, recursively sample the most likely spike, update the posterior, and repeat until the posterior stops increasing. Template matching, projection pursuit regression [32], and matching pursuit [33] are examples of such a greedy approach (Greenberg et al's algorithm could also be considered a special case of such a greedy approach). Both the greedy methods, and the one developed here, aim to optimize a similar objective function. While greedy methods reduce the computational burden by restricting the search space of spike trains, here analytic approximations are made. The advantage of the greedy approaches relative to this one is that they result in a spike train (ie, a binary sequence), whereas the approach developed herein is guaranteed to be approximately optimal, given the model.

Because this filter is model based, it can be generalized in several ways to improve accuracy. Unfortunately, some of these generalizations do not improve inference accuracy, probably because of the exponential approximation. Instead, the fast filter output can be used to initialize the SMC filter [12], to further improve inference quality (c.f. Figure 7). Another model generalization allows incorporation of spatial filtering of the raw movie into this approach (c.f. Figure 8). The parameters of the spatial filter can be estimated from the data, even when spatial filters are overlapping (c.f. Figure 10).

A number of extensions follow from this work. First, further development on some of the model generalizations may improve inference results. Second, pairing this filter with a crude but automatic segmentation tool to obtain ROIs would create a completely automatic algorithm that converts raw movies of populations of neurons into populations of spike trains. Third, combining this algorithm with recently developed connectivity inference algorithms on this kind of data [31], could yield very efficient connectivity inference.

Acknowledgments The authors would like to express appreciation for helpful discussions with Vincent Bonin. Support for JTV was provided by NIDCD DC00109. LP is supported by an NSF CAREER award, by an Alfred P. Sloan Research Fellowship, and the McKnight Scholar Award. RY's laboratory was supported by grant EY11787 from the National Eye Institute and by the Kavli Institute for Brain Science at Columbia University. LP and RY share NSF IIS-0904353.

A Pseudocode

B Wiener Filter

The Poisson distribution can be replaced with a Gaussian instead of a Poisson distribution, ie, $n_t \overset{iid}{\sim} \mathcal{N}(\lambda\Delta, \lambda\Delta)$, which, when plugged into Eq. (7) yields:

$$\widehat{\boldsymbol{n}} = \underset{n_t}{\operatorname{argmax}} \sum_{t=1}^{T} \left(\frac{1}{2\sigma^2}(F_t - \alpha(C_t + \beta))^2 + \frac{1}{2\lambda\Delta}(n_t - \lambda\Delta)^2 \right). \tag{40}$$

Note that since fluorescence integrates over Δ, it makes sense that the mean scales with Δ. Further, since the Gaussian here is approximating a Poisson with high rate [29], the variance should scale with the mean. Using the same tridiagonal trick as above, Eq. (40) can be solved using Newton-Raphson once (because its quadratic). Writing the above in

Algorithm 1 Pseudocode for inferring the approximately most likely spike train, given fluorescence data. Note that $\xi_i \ll 1$ for $i \in \{1, 2, 3\}$; the algorithm is robust to small variations in each; and therefore they were chosen somewhat arbitrarily. The equations listed below refer to the most general equations in the text that one could use (simpler equations could be used when appropriate). Curly brackets, $\{\cdot\}$, indicate comments.

initialize parameters, $\boldsymbol{\theta}$ (section 2.4.1)
while convergence criteria not met **do**
  for $z = 1, 0.1, 0.01, \ldots, \xi_1$ **do** {interior point method to find $\widehat{\boldsymbol{C}}$}
    Initialize $n_t = \xi_2$ for all $t = 1, \ldots, T$, $C_1 = 0$ and $C_t = \gamma C_{t-1} + n_t$ for all $t = 2, \ldots, T$.
    let $\boldsymbol{C}_z$ be the initialized calcium, and $\mathcal{P}_z$, be the posterior given this initialization
    while $\mathcal{P}_{z'} > \mathcal{P}_z + \xi_3$ **do** {Newton-Raphson with backtracking line searches}
      compute $\boldsymbol{g}$ using Eq. (26)
      compute $\boldsymbol{H}$ using Eq. (27)
      compute $\boldsymbol{d}$ using $\boldsymbol{H} \backslash \boldsymbol{g}$ {Gaussian elimination}
      let $\boldsymbol{C}_{z'} = \boldsymbol{C}_z + s\boldsymbol{d}$, where s is between 0 and 1, and $\mathcal{P}_{z'} > \mathcal{P}_z$ {backtracking line search}
    end while
  end for
  check convergence criteria
  update $\vec{\alpha}$ using Eq. (35) {only if spatial filtering}
  update $\vec{\beta}$ using Eq. (36)
  let σ be the root-mean square of the residual
  let $\lambda = \frac{1}{T} \sum_t \widehat{n}_t$
end while

matrix notation, substituting $C_t - \gamma C_{t-1}$ for n_t, yields:

$$\widehat{\boldsymbol{C}} = \underset{\boldsymbol{C}}{\operatorname{argmax}} \frac{1}{2\sigma^2} - \|\boldsymbol{F} - \boldsymbol{C}\|^2 - \frac{1}{2\lambda\Delta} \|\boldsymbol{M}\boldsymbol{C} - \lambda\Delta\boldsymbol{1}\|^2, \tag{41}$$

which is quadratic in $\boldsymbol{C}$. The gradient and Hessian are given by:

$$\boldsymbol{g} = -\frac{1}{\sigma^2}(\boldsymbol{C} - \boldsymbol{F}) - \frac{1}{\lambda\Delta}((\boldsymbol{M}\widehat{\boldsymbol{C}})^{\mathsf{T}}\boldsymbol{M} + \lambda\Delta\boldsymbol{M}^{\mathsf{T}}\boldsymbol{1}), \tag{42}$$

$$\boldsymbol{H} = \frac{1}{\sigma^2}\boldsymbol{I} + \frac{1}{\lambda\Delta}\boldsymbol{M}^{\mathsf{T}}\boldsymbol{M}. \tag{43}$$

Note that this solution is the optimal linear solution, under the assumption that spikes follow a Gaussian distribution, and if often referred to as the Wiener filter, regression with a smoothing prior, or ridge regression [21]. Estimating the parameters for this model follows similarly as described in section 2.4.

References

[1] Rafael Yuste and Arthur Konnerth. *Imaging in Neuroscience and Development, A Laboratory Manual*, 2006.

[2] Diana Smetters, Ania Majewska, and Rafael Yuste. Detecting action potentials in neuronal populations with calcium imaging. *Methods*, 18(2):215–221, Jun 1999.

[3] Yuji Ikegaya, Gloster Aaron, Rosa Cossart, Dmitriy Aronov, Ilan Lampl, David Ferster, and Rafael Yuste. Synfire chains and cortical songs: temporal modules of cortical activity. *Science*, 304(5670):559–564, Apr 2004.

[4] Shin Nagayama, Shaoqun Zeng, Wenhui Xiong, Max L Fletcher, Arjun V Masurkar, Douglas J Davis, Vincent A Pieribone, and Wei R Chen. In vivo simultaneous tracing and Ca^{2+} imaging of local neuronal circuits. *Neuron*, 53(6):789–803, Mar 2007.

[5] Werner Göbel and Fritjof Helmchen. In vivo calcium imaging of neural network function. *Physiology (Bethesda)*, 22:358–365, Dec 2007.

[6] Liqun Luo, Edward M. Callaway, and Karel Svoboda. Genetic dissection of neural circuits. *Neuron*, 57(5):634–660, Mar 2008.

[7] Olga Garaschuk, Oliver Griesbeck, and Arthur Konnerth. Troponin c-based biosensors: a new family of genetically encoded indicators for in vivo calcium imaging in the nervous system. *Cell Calcium*, 42(4-5):351–361, 2007.

[8] Marco Mank and Oliver Griesbeck. Genetically encoded calcium indicators. *Chem Rev*, 108(5):1550–1564, May 2008.

[9] Damian J Wallace, Stephan Meyer zum Alten Borgloh, Simone Astori, Ying Yang, Melanie Bausen, Sebastian Kgler, Amy E Palmer, Roger Y Tsien, Rolf Sprengel, Jason N D Kerr, Winfried Denk, and Mazahir T Hasan. Single-spike detection in vitro and in vivo with a genetic ca2+ sensor. *Nat Methods*, 5(9):797–804, Sep 2008.

[10] David S Greenberg, Arthur R Houweling, and Jason N D Kerr. Population imaging of ongoing neuronal activity in the visual cortex of awake rats. *Nat Neurosci*, Jun 2008.

[11] Terrence F Holekamp, Diwakar Turaga, and Timothy E Holy. Fast three-dimensional fluorescence imaging of activity in neural populations by objective-coupled planar illumination microscopy. *Neuron*, 57(5):661–672, Mar 2008.

[12] Joshua T. Vogelstein, Brendon O. Watson, Adam M. Packer, Rafael Yuste, Bruno Jedynak, and Liam Paninski. Spike inference from calcium imaging using sequential monte carlo methods. *Biophys J*, 97(2):636–655, Jul 2009.

[13] Luis F. Portugal, Joaquim .J. Judice, and Luis N. Vicente. A comparison of block pivoting and interior-point algorithms for linear least squares problems with nonnegative variables. *Mathematics of Computation*, 63(208):625–643, 1994.

[14] Joanne Markham and Jose-Angel Conchello. Parametric blind deconvolution: a robust method for the simultaneous estimation of image and blur. *Journal of The Optical Society Of America A. Optics, Image Science, and Vision*, 16(10):2377–2391, Oct 1999.

[15] Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, Oct 1999.

[16] Yuanqing Lin, Daniel D. Lee, and Lawrence K. Saul. Nonnegative deconvolution for time of arrival estimation. *International Conference on Acoustics, Speech, and Signal Processing*, 2004.

[17] O'Grady, Paul D. and Pearlmutter, Barak A. Convolutive non-negative matrix factorisation with a sparseness constraint. *Machine Learning for Signal Processing, 2006. Proceedings of the 2006 16th IEEE Signal Processing Society Workshop on*, pages 427–432, 2006.

[18] Quentin J. M. Huys, Misha B. Ahrens, and Liam Paninski. Efficient estimation of detailed single-neuron models. *J Neurophysiol*, 96(2):872–890, Aug 2006.

[19] John P. Cunningham, Krishna V. Shenoy, and Maneesh Sahani. Fast Gaussian process methods for point process intensity estimation. *ICML*, pages 192–199, 2008.

[20] Liam Paninski, Yashar Ahmadian, Daniel Ferreira, Shinsuke Koyama, Kamiar Rahnama Rad, Michael Vidne, Joshua Vogelstein, and Wei Wu. A new look at state-space models for neural data. *J Comput Neurosci*, Aug 2009.

[21] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Oxford University Press, 2004.

[22] Emre Yaksi and Rainer W Friedrich. Reconstruction of firing rate changes across neuronal populations by temporally deconvolved Ca^{2+} imaging. *Nature Methods*, 3(5):377–383, May 2006.

[23] Thomas A Pologruto, Ryohei Yasuda, and Karel Svoboda. Monitoring neural activity and $[Ca^{2+}]$ with genetically encoded Ca^{2+} indicators. *J Neurosci*, 24(43):9572–9579, Oct 2004.

[24] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:731, 1997.

[25] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical recipes in C*. Cambridge University Press, 1992.

[26] R.A. Horn and C.R. Johnson. *Matrix analysis*. Cambridge Univ Pr, 1990.

[27] V. Rokhlin, A. Szlam, and M. Tygert. A randomized algorithm for principal component analysis. *SIAM J. Matrix Anal. Appl*, 31:1100–1124, 2009.

[28] Jason N MacLean, Brendon O Watson, Gloster B Aaron, and Rafael Yuste. Internal dynamics determine the cortical response to thalamic stimulation. *Neuron*, 48(5):811–823, Dec 2005.

[29] Lucas Sjulson and Gero Miesenböck. Optical recording of action potentials and other discrete physiological events: a perspective from signal detection theory. *Physiology (Bethesda)*, 22:47–55, Feb 2007.

[30] Christophe Andrieu, Éric Barat, and Arnaud Doucet. Bayesian deconvolution of noisy filtered point processes. *IEEE Transactions on Signal Processing*, 49(1):134–146, 2001.

[31] Mishchenko Y, Vogelstein JT, and Paninski L. A Bayesian approach for inferring neuronal connectivity from calcium fluorescent imaging data. *Annals of Applied Statistics*, in press, 2009.

[32] Jerome H. Friedman and Werner Stuetzle. Projection Pursuit Regression. *J. AM. STAT. ASSOC.*, 76(376):817–823, 1981.

[33] Stephane Mallat and Zhifend Zhang. Matching pursuit with time-frequency dictionaries: IEEE Trans. *Signal Processing*, 41:3397–3415, 1993.