Neuron

Supplemental Information

Simultaneous Denoising, Deconvolution, and Demixing of Calcium Imaging Data

Eftychios A. Pnevmatikakis, Daniel Soudry, Yuanjun Gao, Timothy A. Machado, Josh Merel, David Pfau, Thomas Reardon, Yu Mu, Clay Lacefield, Weijian Yang, Misha Ahrens, Randy Bruno, Thomas M. Jessell, Darcy S. Peterka, Rafael Yuste, and Liam Paninski

Contents

S1 Supplemental Data

S1.1 Supplemental Figures

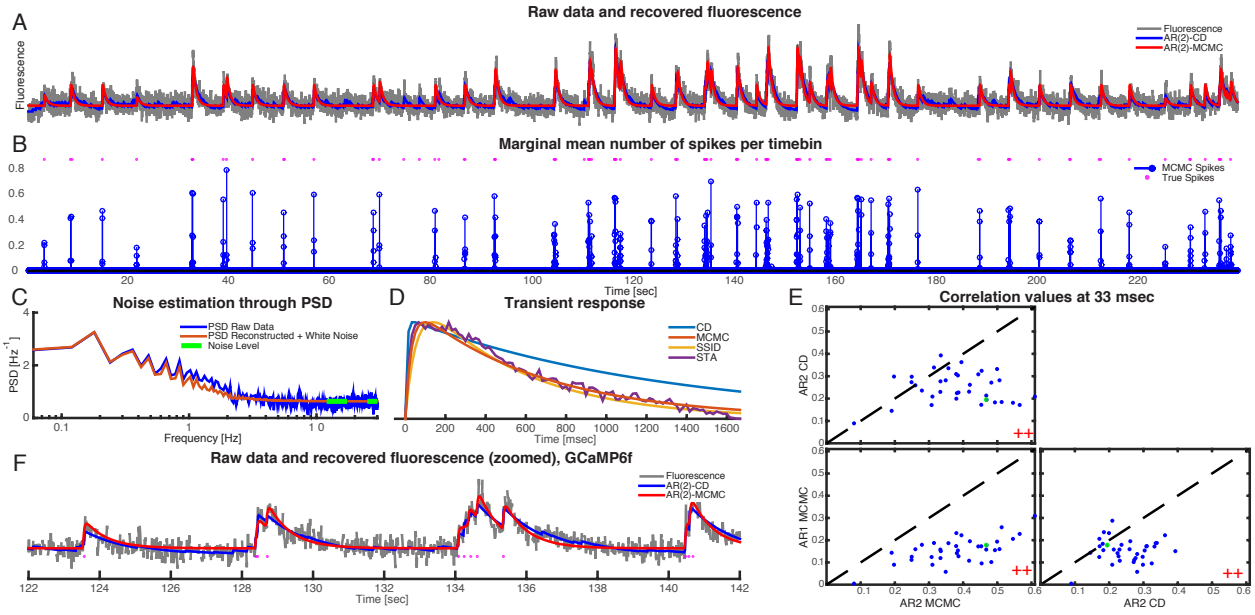

Figure S1, related to Figure 1. Application of the constrained deconvolution algorithm with available ground truth spiking data from the publicly available dataset (GENIE project, Janelia Farm Campus, HHMI; Karel Svoboda (contact), 2015), and illustration of the parameter identification process ($N = 37$ cells). A: Raw fluorescence data and reconstructed traces with constrained deconvolution (blue) and the MCMC (red) methods. B: Reconstructed spike train with MCMC (blue stem plot) and true spikes (purple dots). C: Noise level estimation. The noise level (green) is estimated from the PSD of the raw fluorescence (blue) at high frequencies. The PSD of the reconstructed trace with added white noise of the estimated level (red) matches the PSD of the raw trace. D: Estimation of the spike evoked transient response function with an AR(2) framework. Blue: estimated transient from the sample autocorrelation function (eq. (3) in the main paper). Adaptation of the time constants with the Metropolis-Hastings algorithm within MCMC (red trace). Optimal AR(2) estimate using the ground truth data and systems identification methods (yellow). Normalized spike triggered average (STA) response using the ground truth data (magenta). The sampling methods can approximate the optimal response function as is estimated from the ground truth data. F: Zoomed version of raw and reconstructed data and superimposed spikes. Re-estimation of the time constants improves the modeling of the fluorescence dynamics. E: Correlation scatter plot matrix at 33msec resolution (2$\times$ timebin width) for three different methods: AR(2)-MCMC performs better than AR(2)-CD. Both methods outperform all AR(1) methods (here shown AR(1)-MCMC), establishing that modeling the rise time can significantly improve the quality of deconvolution. Comparisons were done with the Wilcoxon signed-rank test at the 0.05 level.

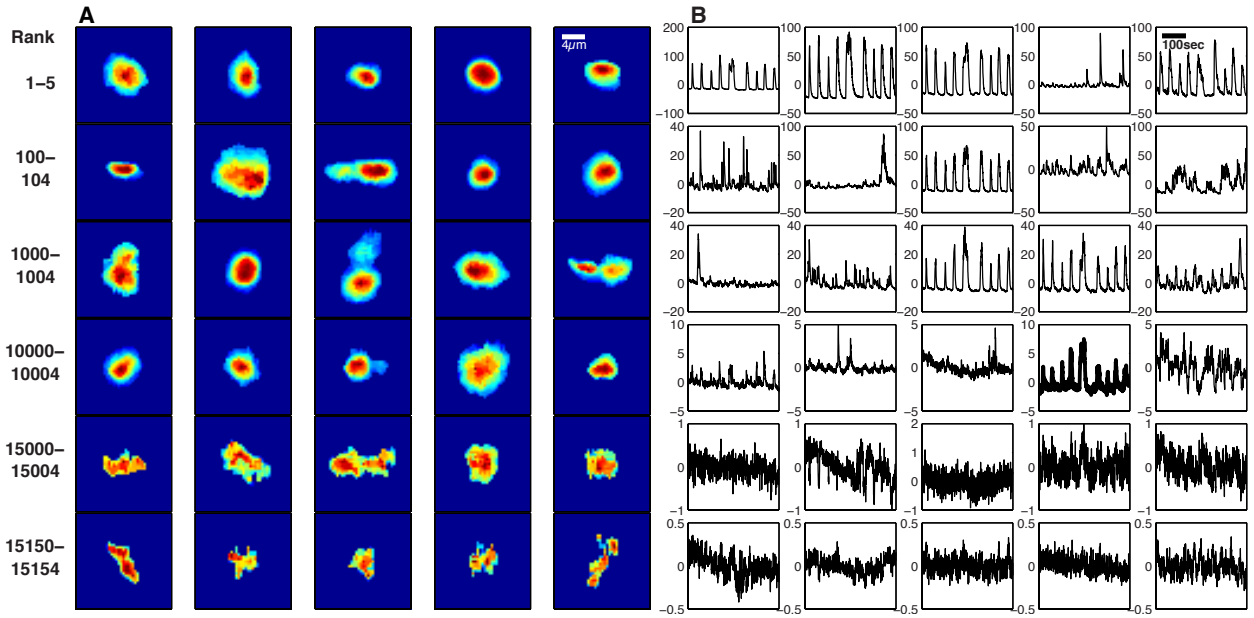

Figure S2, related to Figure 5. Inferred neuronal shapes A_k and DF/F temporal activity using the PCA/ICA algorithm (a subsample from a total of 19013 components detected in whole zebrafish brain). These samples were selected according to the ranking procedure described in the text.

S1.2 Captions for supplementary movies

Movie S1 (related to Figure 4): Application to GCaMP6s-expressing neurons in cortical layer 2/3 of adult mouse V1. Top row: Left: Raw data, Middle: Denoised data with synchronized background activity, Right: Residual signal at 2× finer scale (the synchronized background activity is not included in the residual). Bottom left: Background synchronized activity. For the rest of the panels, 4 representative extracted spatiotemporal components (top) and the corresponding patches of the raw data. The contour plots indicate the locations of the representative components in the field of view. The algorithm successfully denoises the signal and demixes the overlapping neurons.

Movie S2 (related to Figure 6): Video of patch A in Fig. 6, with calcium signals from whole zebrafish recording, obtained with CNMF. Video contains: Raw data, background (**b** from Algorithm S3), denoised video containing only inferred neuronal activity, and the residual (data with Background and inferred neurons removed). Also, we zoom in on components ranked 1st,10th,20th,30th,40th and 50th in each patch (from left to right). For each neuron we show the inferred shape, the centralized data (with background b removed), and the raw data. For the first two, a smaller (occasionally saturated) color scale was used to emphasize very faint (and sometimes briefly firing) neurons, which are also detected by the algorithm.

Movie S3 (related to Figure 6): Video of patch C in Fig. 6, with calcium signals from whole zebrafish recording, obtained with CNMF. Panels similar as in Movie S2.

Movie S4 (related to Figure 6): Video of patch E in Fig. 6, with calcium signals from whole zebrafish recording, obtained with CNMF. Panels similar as in Movie S2.

Movie S5 (related to Figure 6): Video of patch B in Fig. 6, with calcium signals from whole zebrafish recording, obtained with PCA/ICA (Mukamel et al., 2009). Panels similar as in Movie S2.

Movie S6 (related to Figure 6): Video of patch D in Fig. 6, with calcium signals from whole zebrafish recording, obtained with PCA/ICA (Mukamel et al., 2009). Panels similar as in Movie S2. Note only 40 neurons were detected in this patch, so there is no 50th neuron.

Movie S7 (related to Figure 6): Video of patch B in Fig. 6, with calcium signals from whole zebrafish recording, obtained with PCA/ICA (Mukamel et al., 2009). Panels similar as in Movie S2.

Movie S8 (related to Figure 5): Centers of extracted neurons in the zebrafish brain obtained through CNMF. The distribution of visible neurons in the zebrafish whole-brain fluorescence corresponds well with the distribution of the components inferred using constrained NMF (cyan dots). Each frame in the movie is a 2D horizontal slice going upwards in the (dorsal) z direction, with 8 μm spacing. Each pixel in each frame is the 95% percentile of the absolute fluorescence across time.

Movie S9 (related to Figure 7): Application to calcium signals from apical dendrites of cortical Layer 5 pyramidal neurons. Top row: Left: Raw data, Middle: Denoised data with the background and noisy components removed. Right: Residual signal at 2$\times$ finer scale. Bottom panels: 11 of the spatiotemporal extracted components plus the background synchronized activity (lower right panel). The video contains only the frames where at least one of the displayed components is significantly active. The algorithm extracts rich and structured spatiotemporal components that are not visible by plain observation of the raw data.

S2 Supplemental Experimental Procedures

S2.1 Experimental data

Motor spinal neuron data (Fig. 1) Spinal motor neurons expressing the calcium indicator GCaMP6s were imaged in an isolated C57BL/6 mouse spinal cord preparation (aged 4 days postnatal). Ventral roots were stimulated to antidromically evoke patterns of neuronal firing that matched timing of the stimulus pulses (Machado et al., 2015). GCaMP6s expression was achieved following direct spinal cord injections of RV strain SAD-B19 expressing GCaMP6s (Addgene 40753). This vector was constructed and packaged using standard rescue techniques (Osakada et al., 2011). Two-photon imaging (940 nm excitation wavelength, 525/50 emission filter) was conducted at 14.6 Hz using a 20x objective (1.0 N.A; Olympus).

Mouse V1 data (Fig. 4): In vivo calcium imaging data collected from GCaMP6s-expressing neurons in layer 2/3 of the primary visual cortex of an adult mouse. The field of view of the movie was 270μm$\times$270μm and the recording frame rate was 10 fps, and the neurons were imaged through an open skull with a glass cranial window. Expression was achieved via viral injection of AAV1-hsyn-GCaMP6s, four weeks prior to imaging, which was carried out on a home built two-photon laser scanning microscope (see the companion paper, Yang et al. 2015) using a 25X NA 1.05 objective.

Zebrafish light-sheet imaging data (Figs. 5-6): Data obtained as described in Freeman et al. (2014).

Dendritic imaging data (Fig. 7): Calcium signals from apical dendrites of cortical Layer 5 pyramidal neurons were obtained by injecting AAV2/9-hSyn-FLEX-GCaMP6f (UPENN vector core) into the barrel cortex of Rbp4:Cre BAC transgenic mice (GENSAT). Two-photon imaging was performed at 4Hz with a 16x, 0.8NA lens (Nikon) at 940nm while mice performed a whisker-based object detection task (Lacefield and Bruno, 2013, SfN abstract). Resulting TIF stacks were motion corrected with a dynamic programming algorithm implemented in Kaifosh et al. (2014).

S2.2 Algorithmic details

S2.2.1 Parameter estimation and AR modeling

Remember the autoregressive model for the calcium dynamics and the observation noise model

$$\begin{aligned} c(t) &= \sum_{k=1}^{p} \gamma_k c(t-k) + s(t) \\ y(t) &= \alpha(c(t)+b) + \varepsilon(t), \quad \varepsilon(t) \sim \mathcal{N}(0,\sigma^2). \end{aligned} \tag{S1}$$

Under the assumption of a homogeneous Poisson spiking process with $\mathbb{E}[s(t)] = \lambda$, and that the AR process is stationary (it is necessary that $\gamma_1 + \ldots + \gamma_p < 1$), we have

$$\mu \triangleq \mathbb{E}[c(t)] = \frac{\lambda}{1 - \sum_{k=1}^{p} \gamma_k}. \tag{S2}$$

For the auto covariance function C_c we have

$$\begin{aligned} C_c(\tau) &= \mathbb{E}[c(t+\tau)c(t)] - \mu^2 \\ &= \mathbb{E}[\left(\sum_{k=1}^{p} \gamma_k c(t+\tau-k) + s(t+\tau) \right) c(t)] - \mu^2 \\ &= \sum_{k=1}^{p} \gamma_k \mathbb{E}[c(t+\tau-k)c(t)] + \lambda\mu - \mu^2 \\ &= \sum_{k=1}^{p} \gamma_k \left(C_c(\tau-k) + \mu^2 \right) + \left(1 - \sum_{k=1}^{p} \gamma_k \right) \mu^2 - \mu^2 \\ &= \sum_{k=1}^{p} \gamma_k C_c(\tau-k). \end{aligned} \tag{S3}$$

Since the noise is white we have

$$C_y(\tau) = \alpha^2 C_c(\tau) + \sigma^2 \delta(\tau), \tag{S4}$$

and combining (S3) with (S4) we derive

$$C_y(\tau) = \begin{cases} \sum_{k=1}^{p} \gamma_k C_y(\tau-j) - \sigma^2 \gamma_\tau, & 1 \le \tau \le p \\ \sum_{k=1}^{p} \gamma_k C_y(\tau-j), & \tau > p. \end{cases} \tag{S5}$$

The AR coefficients and noise variance can be estimated from the above equations using the sample auto-covariance. Although this method works well for estimating the noise variance in model data, in practice a more robust way to estimate σ^2 is independently by using the power spectral density (PSD) of $\boldsymbol{y}$. This method is somewhat less dependent on parametric model assumptions about the data. Due to the slow decay dynamics of the calcium indicator, the AR process acts typically as a low pass filter on the incoming spikes, and therefore the noiseless calcium trace has very low power in the high frequency range. Since the noise is assumed to be white, its PSD is flat among all frequencies. To estimate σ^2 we compute the PSD of $\boldsymbol{y}$ and then average its value at the range of high frequencies (e.g. in the range $[F_s/4, F_s/2]$), where F_s is the imaging rate. This point is illustrated in more detail in Fig. S1C. A trace from the GCaMP6f dataset presented above is used to illustrate the parameter identification process. The noise level was estimated by averaging the PSD of the raw trace (computed using Welch's method) over a range of high frequencies $[F_s/4, F_s/2]$ (Fig. S1C). The order of the AR process p is typically low (p=1,2), and is determined by the imaging rate. Alternatively, given a noise level, p can be determined as the minimum order such that the deconvolution problem accepts a feasible solution.

S2.2.2 Algorithms for solving the one-dimensional constrained deconvolution problem

We briefly discuss the four different approaches that can be used for the constrained deconvolution problem, which we repeat here for completeness:

$$\begin{aligned} \underset{\boldsymbol{c}}{\text{minimize}} \quad & \mathbf{1}_T^T G(\boldsymbol{c}-\boldsymbol{c}_{\text{in}}), \\ \text{subject to:} \quad & G(\boldsymbol{c}-\boldsymbol{c}_{\text{in}}) \geq 0, \quad c_1 \geq 0 \\ & \|\boldsymbol{y}-\boldsymbol{c}-b\mathbf{1}_T\| \leq \sigma\sqrt{T}. \end{aligned} \tag{P-CD}$$

Here $\boldsymbol{c}_{\text{in}}$ models the initial calcium concentration, a parameter that was omitted in the main text for simplicity. The initial concentration at time $t=1$, is modeled as c_1, and $\boldsymbol{c}_{\text{in}}$ is defined as the vector $\boldsymbol{c}_{\text{in}} = c_1[1,\gamma,\ldots,\gamma^T]^\top$ that models the effect of the initial concentration at the observed time points. The time constant γ models the decay rate of calcium transients initiated before the start of the experiment, as we choose it to be equal to longest decay time constant of the calcium indicator. Using the AR framework, γ is equal to the largest root of the characteristic polynomial $\lambda^p - \gamma_1\lambda^{p-1} - \ldots - \gamma_p = 0$.

For simplicity of the presentation, we assume that the baseline b and initial concentration c_1 are known, although all methods can be trivially modified to include estimation of these parameters. MATLAB implementations for all methods can be found in `https://github.com/epnev/constrained-foopsi`.

Dual ascent methods: We introduce Lagrange multipliers for the constraints and define as $\boldsymbol{c}^\lambda$ as the solution to the following program

$$\begin{aligned} \underset{\boldsymbol{c}}{\text{minimize}} \quad & \mathscr{L}(\boldsymbol{c},\lambda) = \mathbf{1}^T G(\boldsymbol{c}-\boldsymbol{c}_{\text{in}}) + \lambda(\|\boldsymbol{y}-\boldsymbol{c}-b\mathbf{1}_T\|^2 - \sigma^2 T), \\ \text{subject to:} \quad & G(\boldsymbol{c}-\boldsymbol{c}_{\text{in}}) \geq 0. \end{aligned} \tag{S6}$$

The problem (S6) can be readily solved in $O(T)$ time with the interior point method of Vogelstein et al. (2010). After solving (S6), the Lagrange multiplier can be updated as

$$\lambda_k = \lambda_{k-1} - a_k \nabla_\lambda \mathscr{L}(\boldsymbol{c}^{\lambda_{k-1}},\lambda) = \lambda_{k-1} - a_k(\|\boldsymbol{y}-\boldsymbol{c}-b\mathbf{1}_T\|^2 - \sigma^2 T), \tag{S7}$$

where a_k is an appropriate step size, determined e.g. by line search.

Conic programming: The program can also be solved with standard interior point methods for conic programming (Boyd and Vandenberghe, 2004). Due to the simplicity of the residual and non-negativity constraints the solution can be efficiently computed in $O(T)$ time using standard computational methods, e.g. the CVX computational package (Grant et al., 2008).

Nonnegative LARS: The problem can also be solved directly in the spike domain using a nonnegative LARS algorithm (Efron et al., 2004). More specifically we consider the modified problem in the spike domain as follows

$$\begin{aligned} \underset{\boldsymbol{s}}{\text{minimize}} \quad & \frac{1}{2\sigma^2}\|\boldsymbol{y} - G^{-1}\boldsymbol{s} - b\mathbf{1}_T - \boldsymbol{c}_{\text{in}}\|^2 + \lambda\mathbf{1}^T\boldsymbol{s}, \\ \text{subject to:} \quad & \boldsymbol{s} \geq 0. \end{aligned} \tag{S8}$$

Instead of putting a hard noise constraint as before the LARS algorithm computes the solution path of (S8) for all λ and stops when the noise constraint is satisfied. This process is efficient since the solution is piecewise linear in λ. We start from $\lambda_0 = \infty$ where $\boldsymbol{s} = \mathbf{0}_T$. As λ decreases, more spikes are added in the solution, reducing the l_2-norm of the residual signal. Given the solution at the $(k-1)$-th step, at the k-th step, the algorithm includes an additional spike (or removes an existing one), and then optimizes over the spike heights. The path algorithm is stopped when the produced solution satisfies the residual constraint with equality, with a total computational cost $O(TN^2 + N^3)$, where N is the total number of steps. Note that the total number of nonzero spikes is at most k at the k-th step, and therefore $\leq N$ upon termination. Thus this method is particularly efficient and preferred when the spiking signal is expected to be highly sparse.

Spectral projected gradient methods: This method relies on the observation that the curve that characterizes the trade-off between the l_2 norm of the residual ($\|\boldsymbol{y}-\boldsymbol{c}-b\mathbf{1}_T\|$), and the sum of the spiking signal ($\mathbf{1}_T^\top \boldsymbol{s}$), is convex, and uses an interior point method to explore this curve until the desired noise constraint is satisfied. More information can be found in van den Berg and Friedlander (2008). For our case, this method is highly efficient due to our ability to perform fast matrix vector operations with the banded and Toeplitz matrix G, because of the autoregressive model for the calcium dynamics.

S2.2.3 Continuous time interpretation of AR models

We discuss briefly the continuous time interpretation of our autoregressive framework to connect the AR coefficients with some biophysical properties of the calcium indicators. The following discussion is fairly standard, and a more thorough exposition can be found in any standard linear systems textbook (e.g. Oppenheim and Willsky, 1997). An autoregressive model of order p can be written as a discrete time linear dynamical system as follows

$$\underbrace{\begin{bmatrix} c[n] \\ c[n-1] \\ \vdots \\ c[n-p+1] \end{bmatrix}}_{\boldsymbol{c_d}[n]} = \underbrace{\begin{bmatrix} \gamma_1 & \gamma_2 & \dots & \gamma_p \\ 1 & 0 & \dots & 0 \\ & \ddots & \ddots & \vdots \\ 0 & \dots & 1 & 0 \end{bmatrix}}_{A_d} \underbrace{\begin{bmatrix} c[n-1] \\ c[n-2] \\ \vdots \\ c[n-p] \end{bmatrix}}_{\boldsymbol{c_d}[n-1]} + \underbrace{\begin{bmatrix} s[n] \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{\boldsymbol{s}[n]}. \tag{S9}$$

This can be mapped to the following continuous time dynamical system

$$\dot{\boldsymbol{c}}_{\boldsymbol{c}}(t) = A_c \boldsymbol{c}_{\boldsymbol{c}}(t) + \frac{1}{h(\Delta)} \boldsymbol{s}(t), \tag{S10}$$

with

$$\begin{aligned} A_d &= \exp(A_c \Delta), \\ s(t) &= \frac{1}{h(\Delta)} \sum_n s[n] \delta(t - n\Delta), \end{aligned} \tag{S11}$$

where Δ denotes the timebin width, $\delta(\cdot)$ the Dirac delta function, and $h(\cdot)$ is the Green's function of (S10). The eigenvalues of A_d are given by the roots of the characteristic polynomial $\lambda^p - \gamma_1 \lambda^{p-1} - \ldots - \gamma_p = 0$ and the process is stable if all the eigenvalues are within the unit circle. We can apply this framework to model calcium transients with finite rise-time of the form

$$h(t) = \begin{cases} e^{-t/\tau_d} - e^{-t/\tau_r}, & t > 0 \\ 0, & t \le 0 \end{cases}, \tag{S12}$$

with an AR(2) process. The function $h(t)$ is the solution of the continuous time differential equation

$$\underbrace{\begin{bmatrix} \ddot{c}(t) \\ \dot{c}(t) \end{bmatrix}}_{\dot{\boldsymbol{c}}_{\boldsymbol{c}}(t)} = \underbrace{\begin{bmatrix} -\left(\frac{1}{\tau_d} + \frac{1}{\tau_r}\right) & -\frac{1}{\tau_d \tau_r} \\ 1 & 0 \end{bmatrix}}_{A_c} \underbrace{\begin{bmatrix} \dot{c}(t) \\ c(t) \end{bmatrix}}_{\boldsymbol{c}_{\boldsymbol{c}}(t)} + \begin{bmatrix} \delta(t) \\ 0 \end{bmatrix}. \tag{S13}$$

If λ_1^c, λ_2^c are the eigenvalues of A_c, and $\lambda_1^d = \exp(\lambda_1^c \Delta), \lambda_2^d = \exp(\lambda_2^c \Delta)$, then $\gamma_1 = \lambda_1^d + \lambda_2^d$ and $\gamma_2 = -\lambda_1^d \lambda_2^d$. The conditions $\tau_d, \tau_r > 0$ imply that $\lambda_1^c, \lambda_2^c < 0 \Rightarrow 0 < \lambda_1^d, \lambda_2^d < 1$ which implies that $0 < \gamma_1 + \gamma_2 < 1$ and $-1 < \gamma_2 < 0$. λ_1^d, λ_2^d are real numbers when $\gamma_1^2 + 4\gamma_2 > 0$, which is the over-damping condition that prohibits oscillatory behavior in the calcium transient response. With this relationship between γ_1, γ_2 and τ_d, τ_r we have an exact mapping between the continuous and discrete time representations in the sense that

$$\boldsymbol{c}_{\boldsymbol{c}}(n\Delta) = \boldsymbol{c}_{\boldsymbol{d}}[n]. \tag{S14}$$

and we also have $h(\Delta) = \sqrt{\gamma_1^2 + 4\gamma_2}$.

S2.2.4 Updating the time constants

As discussed in the main text, it is useful to refine the time constants of the AR process for each neuron. To do this we augmented the MCMC approach of Pnevmatikakis et al. (2013) to also include sampling of the time constants with a Metropolis-Hastings algorithm (Gelman et al., 2003). This method can fine tune the time constants of the indicator response, at a multiplicative computational cost, since the MCMC method by default draws a number of samples to approximate the full posterior distribution of the spike times and other model parameters.

We briefly describe the process for an AR(2) model of the indicator dynamics. For given time constants τ_d, τ_r in the continuous domain, we sample new values from Gaussian distribution $\tau_d^{\text{new}} \sim \mathcal{N}(\tau_d, \sigma_d^2)$, $\tau_r^{\text{new}} \sim \mathcal{N}(\tau_r, \sigma_r^2)$. We also impose a minimum value of $\tau_{\min}$ for the rise time, and a maximum value $\tau_{\max}$ for the decay time and require that $\tau_r \leq \tau_d$. The proposed time constant values $\tau_d^{\text{new}}, \tau_r^{\text{new}}$ give rise to a new calcium trace $\boldsymbol{c}^{\text{new}}$, and are accepted with probability

$$\mathbb{P}(\text{accept } \tau_d^{\text{new}}, \tau_r^{\text{new}}) = \min\left(1, \frac{\exp\left(-\|\boldsymbol{y} - \boldsymbol{c}^{\text{new}} - b\|^2/2\sigma^2\right)}{\exp\left(-\|\boldsymbol{y} - \boldsymbol{c} - b\|^2/2\sigma^2\right)}\right).$$

An algorithmic description of this procedure is depicted in Alg. S1. In the simpler AR(1) case, then $\tau_r = 0$, and we only need to sample τ_d. The method is also easily extendable to higher order models.

Algorithm S1 Neural activity deconvolution with time constant updating

Require: Data $y \in \mathbb{R}^T$, number of samples N_{samples} for MCMC, timebin width Δ, lower and upper bound of time constants $\tau_{\min}, \tau_{\max}$, variance of proposal kernel σ_r^2, σ_d^2.
Estimate γ_1, γ_2 from (S5), and σ^2 through PSD.
Convert γ_1, γ_2 in continuous time τ_r, τ_d
for i = 1:N_{samples} **do**
Draw new samples for $\boldsymbol{s}, \boldsymbol{c}, c_1, b, \sigma^2$ using the MCMC algorithm of Pnevmatikakis et al. (2013).
Update time constants using UPDATETIMECONSTANTS($\boldsymbol{y}, \boldsymbol{c}, \boldsymbol{s}, b, c_1, \tau_r, \tau_d, \sigma^2, \sigma_r^2, \sigma_d^2, \tau_{\min}, \tau_{\max}$)
end for
return all samples
procedure UPDATETIMECONSTANTS($\boldsymbol{y}, \boldsymbol{c}, \boldsymbol{s}, b, c_1, \tau_r, \tau_d, \sigma^2, \sigma_r^2, \sigma_d^2, \tau_{\min}, \tau_{\max}$)
Draw new time constants

$$\begin{bmatrix} \tau_r^{\text{new}} \\ \tau_d^{\text{new}} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \tau_r \\ \tau_d \end{bmatrix}, \begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_d^2 \end{bmatrix}\right), \quad \text{with} \quad \tau_{\min} \leq \tau_r^{\text{new}} \leq \tau_d^{\text{new}} \leq \tau_{\max}.$$

Construct proposed calcium trace $\boldsymbol{c}^{\text{new}}$.
Draw $r \sim U([0,1])$
if $r < \min\left(1, \frac{\exp\left(-\|\boldsymbol{y} - \boldsymbol{c}^{\text{new}} - b\|^2/2\sigma^2\right)}{\exp\left(-\|\boldsymbol{y} - \boldsymbol{c} - b\|^2/2\sigma^2\right)}\right)$ **then**
$\tau_r \leftarrow \tau_r^{\text{new}}, \ \tau_d \leftarrow \tau_d^{\text{new}}$
end if
return τ_r, τ_d
end procedure

S2.2.5 Merging of existing components

Depending on the initialization procedure, a neuron can sometimes be initially split into two or more different spatial components that subsequently need to be merged. To detect such components, we construct a graph where each vertex corresponds to a neuron and two neurons are connected with an edge if their spatial components overlap. For this graph we detect all the maximal cliques, i.e., the cliques of the graph that are not part of larger cliques. (Fast approximate

methods effective for large sparse graphs exist for this problem (Eppstein et al., 2010).) Now for each of these maximal cliques we compute the correlation matrix of the temporal components of the corresponding nodes. We find the largest principal submatrix where all the correlation coefficients are above a certain threshold, and merge the corresponding spatial components.

S2.2.6 Description of the initialization procedures in somatic imaging

Algorithm S2 Greedy neuron identification

Require: Data $Y \in \mathbb{R}^{d\times T}$; number of neurons needed K; standard deviation of the 2-d Gaussian kernel used for filtering $\tau = (\tau_x, \tau_y)$; window size $w = (w_x, w_y)$.

procedure GREEDYNEURONID(Y, K, τ, w)

$R = Y$;

Define Gaussian blur matrix $D \in \mathbb{R}^{d\times d}$, where column i is a (vectorized) truncated 2-d Gaussian kernel centered at pixel i with variance (τ_x^2, τ_y^2), supported in a $w_x \times w_y$ window centered at i $(1 \le i \le d)$;

for $i = 1 : d$ **do**

Subtract and store median value for each pixel, $m(i) = \text{Median}(Y(i,:))$.

end for

for $k = 1 : K$ **do**

Calculate variance explained by each kernel, $\rho = D^T R$, $v_i = \sum_{t=1}^{T} \rho_{it}^2$;

Identify the center of neuron k, $i_k = \arg\max_i v_i$

Define S_k to be the set of all pixels lie in the $w_x \times w_y$ window centered at i_k; solve by alternating least squares

$$\begin{aligned} \underset{\boldsymbol{a}_k\in\mathbb{R}^d, \boldsymbol{c}_k\in\mathbb{R}^T}{\text{minimize}} \quad & \|R - \boldsymbol{a}_k \boldsymbol{c}_k^T\|^2 \\ \text{subject to:} \quad & a_k(i) \ge 0, i \in S_k \\ & a_k(i) = 0, i \notin S_k. \end{aligned} \tag{S15}$$

$R(S_k) \leftarrow R(S_k) - \boldsymbol{a}_k \boldsymbol{c}_k^T$;

end for

$R \leftarrow R + \boldsymbol{m}\boldsymbol{1}_T^T$. Add median values back to the residual and solve by alternating least squares

$$\begin{aligned} \underset{\boldsymbol{b}\in\mathbb{R}^d, \boldsymbol{f}\in\mathbb{R}^T}{\text{minimize}} \quad & \|R - \boldsymbol{b}\boldsymbol{f}^T\|^2 \\ \text{subject to} \quad & b(i) \ge 0,\ i = 1,\dots,d \\ & f(t) \ge 0,\ t = 1,\dots,T. \end{aligned} \tag{S16}$$

return $A = [\boldsymbol{a}_1, ..., \boldsymbol{a}_K]$, $C = [\boldsymbol{c}_1, ..., \boldsymbol{c}_K]^T$, $\boldsymbol{b}$, $\boldsymbol{f}$.

end procedure

Algorithmic description for the greedy initialization procedure: Algorithm S2 provides full details of the greedy initialization procedure, which was described at a high level in the methods section. Several details are worth noting here. To begin we center the data at each pixel around zero, by subtracting the median over time. Also, since the spatial component S_k is localized, at each step only a small portion of the residual is updated and therefore only a small portion of the explained variance needs updating. Also note that the nonnegative matrix factorization step (equation (S15) in algorithm S2) can be efficiently done by alternating between optimizing $\boldsymbol{a}_k$ and $\boldsymbol{c}_k$, since the Gaussian kernel scan gives a reasonable initialization. Usually $\sim$5 iterations are enough for convergence. Since the solution of (S15) is identifiable up to a multiplicative scalar, we constrain the spatial components to have unit norm.

Algorithmic description for the Group Lasso initialization: In order to solve the convex optimization problem of eq. (8), we use the FISTA algorithm (Algorithm S3, derived from Eq. 4.1-4.3 on Beck and Teboulle (2009)), where $R = Y - \boldsymbol{b}\mathbf{1}_T^\top$ is the residual from the data after we subtract the background component. As explained in Beck and Teboulle (2009) a key parameter of the algorithm is the "Lipschitz constant," which in the setting considered here is twice the maximal eigenvalue of $D^\top D$. In our case, we use a filter matrix D whose sum is normalized to 1, and whose Fourier spectrum is strictly decreasing as a function of the magnitude of the frequency, so the maximal eigenvalue is 1. Therefore the Lipschitz constant is $L = 2$.

Next, we list a few implementation details. (1) It usually takes 30-100 iterations of the algorithm to converge. (2) All Gaussian dictionary elements have equal standard deviation τ and window size $w = 4\tau$, chosen to match the typical size of neurons in the image. (3) Convolution with the Gaussian Kernel is implemented efficiently using the Fast Fourier Transform. This is especially fast if graphical processing units (GPUs) are used. (4) The regularization constant λ can be automatically adjusted by requiring that the portion of significant locations (i.e. i in which $\sum_t F_{it}^2 \geq 0$) in the image is equal to the estimated neuronal density in the image area in Algorithm S3). This can be done efficiently using using exponential search (which is an efficient generalization of binary search for unbounded lists). (5) If the neuronal shape detected by the matrix factorization approach has a center which is far (e.g, 2τ) from the original shape detected by the group lasso initialization, then we discard that shape.

Algorithm S3 FISTA for Group Lasso

Require: Residual $R \in \mathbb{R}^{d\times T}$; Initial points standard deviation of the 2-D Gaussian kernel $\tau = (\tau_x, \tau_y)$; Regularization constant λ;

1: **procedure** GROUPLASSOFISTA(R, τ, α,λ)

2: Define Gaussian blur matrix $D \in \mathbb{R}^{d\times d}$, where column p is a (vectorized) truncated 2-D Gaussian kernel centered at p with variance (τ_x^2, τ_y^2), and the operator

$$\left(\mathscr{T}_\mu(a)\right)_{qt} \triangleq \max\left[a_{qt}\left(1 - \frac{\mu}{\sqrt{\sum_t a_{qt}^2}}\right), 0\right].$$

3: Initialize $\mu = \lambda/L$, $L = 2$, $F_{(0)} = 0^{d\times T}$, ,$W_{(1)} = F_{(0)}$, $t_{(1)} = 1$, $M = \frac{2}{L}D^\top D$, $v = \frac{2}{L}D^\top R$.

4: **Repeat** until convergence for $k \geq 1$:

5: FISTA equations:

$$\begin{aligned} F_{(k)} &= \mathscr{T}_\mu\left[W_{(k)} - MW_{(k)} + v\right] \\ t_{(k+1)} &= \left(1 + \sqrt{1 + 4t_{(k)}^2}\right)/2 \\ W_{(k+1)} &= F_{(k)} + \left(\frac{t_{(k)} - 1}{t_{(k+1)}}\right)\left(F_{(k)} - F_{(k-1)}\right) \end{aligned}$$

return $F, \boldsymbol{b}$.

6: **end procedure**

S2.2.7 Handling missing data

Our method can easily handle the case of missing data that can arise in practice, e.g., due to brain movement during line scanning (Dombeck et al., 2007). Our framework remains the same, with the only difference that entries where observations are missing are omitted from the noise constraints, but are included in the constraints for the temporal dynamics. To compute the autocovariance function or PSD (for estimating the AR parameters and noise level) for a pixel with missing data, we interpolate the missing values using nearest neighbor interpolation

	Imaging rate	AR order	Initialization method	Include noise constraints
Somatic or axonal imaging	$f \geq 15$Hz	$p = 2$	Greedy/GL	Yes
	$5\text{Hz} \leq f \leq 15\text{Hz}$, slow indicator	$p = 1,2$	Greedy/GL	Yes
	$5\text{Hz} \leq f \leq 15\text{Hz}$, fast indicator	$p = 1,2$	Greedy/GL	Yes
	$f \leq 5$Hz	$p = 0$ (no deconvolution)	Greedy/GL	No
Dendritic Imaging	$f \geq 10$Hz	$p = 0$ (no deconvolution)	Sparse NMF	Yes
	$f \leq 5$Hz	$p = 0$ (no deconvolution)	Sparse NMF	No

Table S1, related to the Experimental Procedures. Recommended algorithm setting depending on imaged modality, imaging rate and calcium indicator.

S2.2.8 Extraction of DF/F values

Our approach enables us to express the temporal trace of each component in the DF/F domain, independently of the relative amplitude between the spatial and temporal components. To do so, we can assign a temporal background signal to each component, by averaging the spatiotemporal background over the spatial component:

$$\mathbf{f}_j^0 = (\mathbf{a}_j^\top (Y - AC))^\top,$$

and then express the DF/F values as $\mathbf{c}_j / m(\mathbf{f}_j^0)$, where $m(\cdot)$ is an appropriate summary statistic (e.g., mean, median, running average).

S2.2.9 Further algorithmic speedups

Further algorithmic speedups can be obtained during the initialization phase, by spatially downsampling the raw to obtain faster initial estimates and using specific block coordinate descent strategies during the CNMF iterations. Such approaches can lead to up to an order of magnitude speedups as highlighted in Friedrich et al. (2015) on zebrafish light-sheet imaging data.

S2.3 Details of the data analysis

Our proposed method is modular and comes with multiple different variants that can be used during the initialization and/or the alternating minimization process. Before we present the details for all the datasets and examples presented in the main paper, Table S1, related to the Experimental Procedures presents some general guidelines on what variant to use depending on the imaged modality, imaging rate and/or calcium indicator used.

S2.3.1 Details of the application to spinal cord data of Fig. 1

The constrained deconvolution problem (P-CD) was solved with conic programing using the CVX computational package. For updating the time constants, the standard deviations for the proposal density were chosen as $\sigma_d = 10$msec, $\sigma_r = 2$msec. Furthermore, the bounds were set as $\tau_{\min} = 0$, $\tau_{\max} = 2000ms$. For the MCMC algorithm, $N_{\text{samples}} = 500$ samples were drawn as described in Pnevmatikakis et al. (2013).

S2.3.2 Details of the simulated experiment of Fig. 2

The field of view was 50×50 pixels large. The two neurons had a 2-d spherical Gaussian shape with standard deviation 5 pixels and were centered on the same horizontal axis, 3 pixels apart, giving a high overlap between the two spatial footprints (correlation value = 0.9). Spikes were simulated from a Bernoulli process with probability of spiking per timebin 0.05. 2000 timebins were simulated, and the calcium was generated from a first order process with $\gamma = 0.8$, corresponding to a decay time constant 4.48 times the width of the timebin. We used a simple clustering procedure to initialize the estimates: after an iteration of the sparse constrained NMF, we clustered the extracted spiking signals according to a simple max-assignment

$$s_1^{\text{new}}(t) = \begin{cases} s_1^{\text{old}}(t) + s_2^{\text{old}}(t), & s_1^{\text{old}}(t) > s_2^{\text{old}}(t) \\ 0, & \text{otherwise} \end{cases},$$

and similarly for $s_2^{\text{new}}(t)$. These clustered temporal components were then used to warm-start the constrained NMF.

S2.3.3 Details of the simulated experiment of Fig. 3

The field of view was 50×50 pixels large. Simulations were performed with neurons that had two different shapes: (i) spherical 2-d Gaussians with standard deviation 5 pixels and (ii) "donut" shaped centered at $[x_0, y_0]$ with shape given by

$$\alpha(x,y) = \exp\left(-\frac{(\sqrt{(x-x_0)^2 + (y-y_0)^2} - r_0)^2}{2\sigma_r^2}\right),$$

with $r_0 = 4, \sigma_r = 1$. In each simulation 10 neurons were placed in the field with centers drawn from a spatial Poisson process with intensity function at each pixel chosen form a uniform distribution. This setup allowed for arbitrary spatial patterns and degree of overlap. Spikes were simulated from a Bernoulli process with probability of spiking per timebin 0.05. 2000 timebins were simulated, and the calcium was generated from a first order process with $\gamma = 0.9$, corresponding to a decay time constant 9.49 times the width of the timebin. White Gaussian noise was simulated to corrupt the data. The standard deviation for each pixel was equal to $\xi \times$ the mean activity, and thirty different noise levels were considered, $\xi = 0.1, 0.2, \ldots, 3$. For each combination of noise level and neuron shape 5 simulations were performed. The plain NMF of Maruyama et al. (2014) searched for 10 neurons, with the temporal background fixed as described in Maruyama et al. (2014). For our constrained NMF framework, we used the greedy method to initialize the 10 components, and estimated the noise level and time constant from the data. To estimate the spiking signal for the PCA/ICA and plain NMF methods, the estimated temporal components were deconvolved from the true indicator dynamics (giving these methods a bit of a relative advantage). Fig. 3 reports the median spike correlation values among all 5 trials and 10 estimated neurons.

S2.3.4 Details of the application to the large scale V1 data (Fig. 4)

The greedy initialization algorithm was used to initialize 300 spatio-temporal components with size of filtering kernel 4×4 and window 10×10. This number was chosen after visual inspection of the raw data. Then the CNMF framework was applied and the constrained deconvolution algorithm was used for estimating the temporal components. Since the imaging rate was relatively low (10Hz), an AR(1) process was used to model the temporal traces, and each component had a separate time constant that was estimated from the CD algorithm. At the end of each iteration, the merging procedure was applied with a merging threshold of 0.8. In total 31 components were merged. The remaining 269 components had time constants that are within a factor of two of the values reported in the literature for GCaMP6s (Chen et al., 2013), with a median value of 1180ms. The temporal traces were then transformed into the DF/F domain, and ordered according to the criterion explained in the Experimental Procedures.

S2.3.5 Details of the application to the light-sheet imaging data (Figs. 5-6)

The data is segmented to 5600 patches, and the algorithm is then run in parallel on all patches. In order to determine the value of the regularization parameter λ used for the group lasso initialization approach (8), we sample a few patches out of the whole data set, and find a single value of λ which gives reasonable results for all these patches. We then use this single value of λ to provide initial group lasso estimates for the whole brain. To prevent the detection of partial shape components we ignore any shape detected near the edge of the patch, and use overlapping patches to compensate. To do so, we remove any shape whose activity is highly correlated (above 0.95) to some other higher-ranked overlapping shape component.

Since the spatial components are restricted to lie within their corresponding spatial patches (which are not large compared to the size of the cell body), the inferred spatial components are localized (and therefore sparse) by construction. Thus there is no need to impose a spatial sparsity constraint. Similarly, because of the low temporal resolution of these recordings, the inferred neural activity vectors are not expected to be particularly sparse, and therefore we do not impose sparsity in the temporal domain either. This leads to a somewhat simplified optimization problem:

$$\begin{aligned} \underset{A,C,\boldsymbol{b},\boldsymbol{f}}{\text{minimize}} \quad & \|Y - \boldsymbol{b}\boldsymbol{f}^\top - AC\|^2 \\ \text{subject to} \quad & A_{xk} \geq 0,\ \forall x \in S_k \\ & A_{xk} = 0,\ \forall x \notin S_k \\ & C, \mathbf{b}, \mathbf{f} \geq 0 \end{aligned} \tag{S17}$$

where S_k denotes the k-th fixed spatial patch. We solve this problem by block-coordinate descent; upon convergence, we lightly smooth the inferred shape using median filtering. Because the resulting spatial components here were not quite as sparse as in the other examples (because we did not sparsen A within the patches S_k), we found it useful when ranking the obtained components to multiply the maximum value of the temporal components by the squared l_4 norm of the corresponding spatial footprints, to penalize overly broad and/or noisy spatial shapes.

S2.4 Application of constrained calcium deconvolution to datasets with available ground truth (Fig. S1)

We applied the constrained deconvolution algorithm to two publicly available datasets (GENIE project, Janelia Farm Campus, HHMI; Karel Svoboda (contact), 2015) with available ground truth. Prior to deconvolution the data was high pass filtered with a quantile filter that subtracted the 10th percentile value over a moving window with width 8.3sec (500 timebins, imaging rate 60Hz) to remove slowly variable baseline due to background/neuropil activity. Due to the high imaging rate, an AR(2) process was used to model both the rise and the decay of the calcium indicator spike triggered response function. The results are shown in Fig. S1. The benefits that can be derived from this time constant updating scheme are shown in Fig. S1D. The AR estimation method (eq. S5, blue trace) estimates $\tau_r = 8$msec and $\tau_d = 1277$msec, a much faster rise and slower decay than trace recovered if we compute the spike triggered average response (magenta trace), and does not the fit data well (Fig. S1F). These estimates are corrected by the MCMC (red) approach, $\tau_r = 32$msec, $\tau_d = 645$msec, respectively. These traces match better the STA response. For reference applying the (supervised) n4sid systems identification method (Verhaegen and Verdult, 2007) to estimate a second order model gives $\tau_r = 62$msec, $\tau_d = 518$msec (yellow).

References

Beck A., and Teboulle M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM Journal on Imaging Sciences *2*, 183–202.

Boyd S., and Vandenberghe L. (2004). Convex Optimization (Oxford University Press).

Chen T.W., Wardill T.J., Sun Y., Pulver S.R., Renninger S.L., Baohan A., Schreiter E.R., Kerr R.A., Orger M.B., Jayaraman V., et al. (2013). Ultrasensitive fluorescent proteins for imaging neuronal activity. Nature *499*, 295–300.

Dombeck D.A., Khabbaz A.N., Collman F., Adelman T.L., and Tank D.W. (2007). Imaging large-scale neural activity with cellular resolution in awake, mobile mice. Neuron *56*, 43–57.

Efron B., Hastie T., Johnstone I., Tibshirani R., et al. (2004). Least angle regression. Ann Statist *32*, 407–499.

Eppstein D., Löffler M., and Strash D. (2010). Listing all maximal cliques in sparse graphs in near-optimal time. In Algorithms and Computation (Springer), vol. 6506 of *Lecture Notes in Computer Science*, pp. 403–414.

Freeman J., Vladimirov N., Kawashima T., Mu Y., Sofroniew N.J., Bennett D.V., Rosen J., Yang C.T., Looger L.L., and Ahrens M.B. (2014). Mapping brain activity at scale with cluster computing. Nat Methods *11*, 941–950.

Friedrich J., Soudry D., Mu Y., Freeman J., Ahres M., and Paninski L. (2015). Fast constrained non-negative matrix factorization for whole-brain calcium imaging data. In NIPS workshop on statistical methods for understanding neural systems.

Gelman A., Carlin J., Stern H., and Rubin D. (2003). Bayesian Data Analysis (CRC Press).

GENIE project, Janelia Farm Campus, HHMI; Karel Svoboda (contact) (2015). Genie project, janelia farm campus, hhmi. http://dx.doi.org/10.6080/K02R3PMN.

Grant M., Boyd S., and Ye Y. (2008). CVX: Matlab software for disciplined convex programming.

Kaifosh P., Zaremba J.D., Danielson N.B., and Losonczy A. (2014). SIMA: Python software for analysis of dynamic fluorescence imaging data. Front Neuroinform *8*.

Machado T., Pnevmatikakis E., Paninski L., Jessell T., and Miri A. (2015). Primacy of flexor locomotor pattern revealed by ancestral reversion of motor neuron identity. Cell *162*, 338–350.

Maruyama R., Maeda K., Moroda H., Kato I., Inoue M., Miyakawa H., and Aonishi T. (2014). Detecting cells using non-negative matrix factorization on calcium imaging data. Neural Netw *55*, 11–19.

Mukamel E.A., Nimmerjahn A., and Schnitzer M.J. (2009). Automated analysis of cellular signals from large-scale calcium imaging data. Neuron *63*, 747–760.

Oppenheim A.V., and Willsky A.S. (1997). Signals and systems (Prentice-Hall).

Osakada F., Mori T., Cetin A.H., Marshel J.H., Virgen B., and Callaway E.M. (2011). New rabies virus variants for monitoring and manipulating activity and gene expression in defined neural circuits. Neuron *71*, 617–631.

Pnevmatikakis E., Merel J., Pakman A., and Paninski L. (2013). Bayesian spike inference from calcium imaging data. In Asilomar Conference on Signals, Systems & Computers. pp. 349–353.

van den Berg E., and Friedlander M.P. (2008). Probing the pareto frontier for basis pursuit solutions. SIAM Journal on Scientific Computing *31*, 890–912.

Verhaegen M., and Verdult V. (2007). Filtering and system identification: a least squares approach (Cambridge Univ Pr).

Vogelstein J., Packer A., Machado T., Sippy T., Babadi B., Yuste R., and Paninski L. (2010). Fast non-negative deconvolution for spike train inference from population calcium imaging. J Neurophysiol *104*, 3691–3704.