A structured matrix factorization framework for large scale calcium imaging data analysis

Eftychios A. Pnevmatikakis Yuanjun Gao Daniel Soudry David Pfau
Clay Lacefield Kira Poskanzer Randy Bruno Rafael Yuste
Liam Paninski
Columbia University

Abstract

We present a structured matrix factorization approach to analyzing calcium imaging recordings of large neuronal ensembles. Our goal is to simultaneously identify the locations of the neurons, demix spatially overlapping components, and denoise and deconvolve the spiking activity of each neuron from the slow dynamics of the calcium indicator. The matrix factorization approach relies on the observation that the spatiotemporal fluorescence activity can be expressed as a product of two matrices: a spatial matrix that encodes the location of each neuron in the optical field and a temporal matrix that characterizes the calcium concentration of each neuron over time. We present a simple approach for estimating the dynamics of the calcium indicator as well as the observation noise statistics from the observed data. These parameters are then used to set up the matrix factorization problem in a constrained form that requires no further parameter tuning. We discuss initialization and post-processing techniques that enhance the performance of our method, along with efficient and largely parallelizable algorithms. We apply our method to *in vivo* large scale multi-neuronal imaging data and also demonstrate how similar methods can be used for the analysis of *in vivo* dendritic imaging data.

1 Introduction

Calcium imaging is becoming a standard tool for monitoring large neuron populations. Over the recent years exciting developments have enabled whole brain imaging of small animals (Ahrens et al., 2013; Prevedel et al., 2014) at reasonable imaging rates. On a different front, engineering of genetically encoded calcium indicators continues, offering increasingly sensitive indicators that can reliably detect single action potentials in *in vivo* conditions (Chen et al., 2013). These developments pose significant challenges from a statistical viewpoint. The data analyst typically faces three major problems: (i) identifying the region of interest (ROI) of each neuron in the optical field, (ii) demixing spatially overlapping ROIs (where overlap is due either to the projection of a $3d$ volume onto a $2d$ imaging plane, or to insufficient spatial resolution in $3d$ imaging methods) and (iii) deconvolving (and denoising) the spiking activity of each neuron from the much slower dynamics of the calcium indicator.

These problems have been traditionally treated independently in the literature. Methods of spike deconvolution have focused largely on single-pixel fluorescence data analysis. Such methods include fast nonnegative deconvolution (Vogelstein et al., 2010), greedy algorithms (Grewe et al., 2010), finite rate of innovation methods (Oñativia et al., 2013), as well as particle filtering (Vogelstein et al., 2009) and MCMC methods (Pnevmatikakis et al., 2013). While effective in the analysis of single fluorescence traces, these methods do not address the problem in a full spatiotemporal setup.

ROI identification is usually based on either of the following two observations: first, ROIs are often spatially localized, yielding methods for ROI selection based on local correlations of neighboring pixels (Smith and Häusser, 2010) or dictionary learning (Pachitariu et al., 2013). While these methods can yield compact localized ROI estimates they do not exploit the spatiotemporal data optimally and their performance can deteriorate in the case of significant spatial overlap. A different approach stems from the observation that the spatiotemporal activity can be expressed as a product of two matrices: a spatial matrix that encodes the location of each neuron in the optical field, and a temporal matrix that characterizes the calcium concentration evolution of each neuron. Based on this observation several methods have been proposed based on independent component analysis (Mukamel et al., 2009), multilevel sparse matrix factorization (Andilla and Hamprecht, 2013), and constrained nonnegative matrix factorization (Maruyama et al., 2014). These methods can deal more effectively with overlapping sources but again do not explicitly model the calcium indicator dynamics and do not necessarily provide compact ROI estimates.

In this paper we approach all three problems simultaneously, by proposing a constrained matrix factorization method that decomposes the spatiotemporal activity into spatial and temporal components that model the dynamics of the calcium indicator and preserve the local structure of each ROI. Related methods have appeared recently in the literature. In Pnevmatikakis et al. (2013) and Pnevmatikakis and Paninski (2013) the authors propose a rank-penalized approach to initialize a matrix factorization algorithm that enforces the calcium indicator dynamics and penalizes the sparsity of each component. More recently, Haeffele et al. (2014) propose a similar structured matrix factorization approach with a spatial total variation norm penalty to promote localized and compact ROIs. While these approaches are effective in real data analysis, they require tuning of several regularization weights, a task that can be very challenging in practice.

In our method we address this problem by proposing a constrained and structured matrix factorization approach that requires no tuning of sparsity parameters. We achieve this by introducing for each pixel a hard constraint on the energy of the residual signal between the raw data and the denoised calcium signal. These hard threshold noise levels can be estimated by exploiting the autoregressive structure of the calcium indicator dynamics. The resulting matrix factorization approach enforces the dynamics of the calcium indicator and effectively sets individual sparsity penalties for both the spiking activity and the ROI size of each neuron that are optimally tuned to satisfy the estimated residual constraints.

We present algorithms for solving this matrix factorization problem that, per iteration, scale linearly both with the total number of observed pixels and the number of timesteps, and present warm start methods that increase the computational efficiency. We also show how the methods can be parallelized to a large extent, leading to a highly efficient system that can process large movies within just a few minutes. Matrix factorization methods typically solve a bi-convex problem and their performance depends on the initialization; we propose a fast and simple greedy initialization method that detects possible neuron locations using spatial filtering methods. Finally, we introduce a few additional processing steps within the matrix factorization procedure that enhance the robustness of the algorithm and make it less sensitive to poor initialization.

We apply our method to an *in vivo* large scale calcium imaging dataset and demonstrate excellent performance. Finally, to demonstrate the generality of modern matrix factorization methods in the analysis of calcium imaging data, we apply a simplified version of our algorithm to dendritic imaging data and show how such methods can effectively segment complex and dense imaging datasets.

2 Optimal constrained deconvolution for single pixel fluorescence traces

For ease of exposition we first address the problem of spike deconvolution from a single-pixel fluorescence times series that expresses the behavior of a single neuron. The calcium dynamics $\boldsymbol{c}$ can be approximated by a stable autoregressive process of order p (AR(p)) where p is a small positive integer,

$$c(t) = \sum_{j=1}^{p} \gamma_j c(t-j) + s(t), \tag{1}$$

and $s(t)$ is the spiking signal (i.e., number of spikes) that the neuron fired at the t-th timestep, $t = 1, \dots, T$. The observed fluorescence is related to the calcium concentration as:

$$y(t) = \alpha c(t) + b + \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, \sigma^2),$$

where α is a nonnegative scalar, b is the baseline concentration and the noise is assumed to be i.i.d. zero mean Gaussian with variance σ^2. We assume that the baseline b is known and constant, e.g. it is estimated by averaging the fluorescence over a large interval with no observed spikes. We relax both of these assumptions in the spatiotemporal case. Our goal is to perform spike inference, i.e., extract the spiking vector $\boldsymbol{s}$ from the vector of observations $\boldsymbol{y}$.

To solve this problem we need first to estimate certain parameters. These include the order of the AR process p, the AR coefficients $\gamma_1, \dots, \gamma_p$, and the observation noise variance σ^2. Assuming independent and identically distributed statistics of the spiking signal, these parameters can be estimated from standard time series analysis methods. For a given order p, it is easy to show that the autocovariance function of $\boldsymbol{y}$, C_y satisfies the following equations:

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

By plugging the sample autocovariance values into (2) we can first estimate the AR coefficients $\gamma_1, \dots, \gamma_p$ and then the noise variance σ^2. For the order p in general we note that if the rise time of the calcium indicator is much faster than the length of each timebin then we can safely assume that $p = 1$. Otherwise, the order of the AR system can be estimated with the Akaike information criterion (AIC) (Akaike, 1969). For notational simplicity we assume $p = 1$ in the following.

Spike inference through optimal noise constrained deconvolution: Solving for the spiking vector $\boldsymbol{s}$ in the domain of nonnegative integers is a computationally hard problem. Instead, by following Vogelstein et al. (2010) we can relax the spike signal to take arbitrary nonnegative values and penalize the sum of the spike signal over time to avoid overfitting. For computational reasons it is also preferable to infer the calcium signal $\boldsymbol{c}$ instead of the spiking vector $\boldsymbol{s}$. By introducing c_1, the initial concentration at the first timestep, and $\boldsymbol{c}_{\text{in}}$ the vector of length T given by $\boldsymbol{c}_{\text{in}} = [c_1, 0, \dots, 0]^\top$, (1) can be expressed in matrix form as

$$G(\boldsymbol{c} - \boldsymbol{c}_{\text{in}}) = \boldsymbol{s}, \ \text{ with } \ G = \begin{bmatrix} 1 & 0 & \dots & 0 \\ -\gamma & 1 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & -\gamma & 1 \end{bmatrix}. \tag{3}$$

Based on the estimate of the noise variance we can introduce a hard constraint on the energy of the residual signal to derive the following parameter-free convex program for estimating the calcium concentration up to a scaling constant:

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

with $||.||$ denoting the 2-norm. The inclusion of the residual as a hard constraint and not as a penalty term in the objective function (as in Vogelstein et al. (2010)) allows for a parameter-free fast non-negative deconvolution approach. After solving (P-1d) we can use (3) to obtain the relaxed spiking signal. Program (P-1d) can be solved efficiently with a variety of methods, such as dual ascent (using the FOOPSI algorithm from Vogelstein et al. (2010) in the primal step) or conic programming (using, e.g., the cvx optimization package (Grant et al., 2008)). These methods scale linearly with the total number of timesteps T. Alternatively, we can also solve (P-1d) directly in the spike domain using a nonnegative LARS algorithm (Efron et al., 2004). This LARS approach is particularly efficient when the spiking signal is expected to be very sparse, so the path following algorithm stops only after a few steps. We present details of these different approaches in the supplement.

3 Spatiotemporal spike inference and component demixing through constrained matrix factorization

Now we turn to the full spatiotemporal case. At every timestep a field of view is observed, for a total number of T timesteps. This field (either two- or three-dimensional) has a total number of d pixels and can be vectorized in a single column vector. Thus all the observations can be described by a $d \times T$ matrix Y. Now assume that the field contains a total number of (possibly overlapping) K neurons, where K is assumed known for now. For each neuron i the "calcium activity" $\boldsymbol{c}_i$ can be described again with simple autoregressive dynamics (assumed first-order just for notational simplicity),

$$c_i(t) = \gamma c_i(t-1) + s_i(t), \tag{4}$$

where $s_i(t)$ is the number of spikes that neuron i fired at the t-th timestep, $t = 1, \dots, T$. Now if $\boldsymbol{a}_i \in \mathbb{R}_+^d$ denotes the (nonnegative) spatial "footprint" vector for neuron i, then we model the spatial calcium concentration profile at time t as

$$F(t) = \sum_{i=1}^{K} \boldsymbol{a}_i c_i(t) + B(t), \tag{5}$$

where $B(t) \in \mathbb{R}_+^d$ denotes the (time-varying) baseline vector all the pixels. Finally, at each timestep we observe $F(t)$ corrupted by additive Gaussian noise:

$$Y(t) = F(t) + \boldsymbol{\varepsilon}_t, \quad \boldsymbol{\varepsilon}_t \sim \mathcal{N}(\mathbf{0}, \Sigma), \tag{6}$$

where Σ is a diagonal matrix (indicating that the noise is spatially and temporally uncorrelated). Eqs. (4)-(6) can be written in matrix form as[1]

$$\begin{aligned}S &= CG^\top\\ F &= AC + B\\ Y &= F + E,\end{aligned}$$

with $S = [\boldsymbol{s}_1, \dots, \boldsymbol{s}_K]^\top$, $C = [\boldsymbol{c}_1, \dots, \boldsymbol{c}_K]^\top$, $A = [\boldsymbol{a}_1, \dots, \boldsymbol{a}_K]$, $F = [F(1), F(2), \dots, F(T)]$, $Y = [Y(1), Y(2), \dots, Y(T)]$, $B = [B(1), B(2), \dots, B(T)]$. In practice, we have found that the background activity matrix B can often

[1] We ignore the initial values for simplicity.

be modeled as a rank 1 matrix, $B = \boldsymbol{b}\boldsymbol{f}^\top$, where $\boldsymbol{b} \in \mathbb{R}_+^d$, $\boldsymbol{f} \in \mathbb{R}_+^T$ are nonnegative vectors encoding the background spatial structure (typically consisting of a sum of baseline activity from the neurons of interest and densely mixed neuropil structure below the observed spatial resolution) and global (possibly time varying) intensity, respectively.[2]

3.1 Optimal matrix factorization deconvolution methods

Assuming the number of neurons K and initial estimates of A, C and $\boldsymbol{b}, \boldsymbol{f}$ as known, we can apply alternating matrix factorization methods to estimate the spatial components $A, \boldsymbol{b}$ given the temporal $C, \boldsymbol{f}$ and vice versa, from the fluorescence observations Y. We present an efficient initialization procedure in section 3.2.

This is a sparse decomposition problem

Estimating $A, \boldsymbol{b}$: Since each column of A expresses the location of a neuron, we want A to be sparse to promote localized spatial footprints. Given estimates of C and $\boldsymbol{f}$ from the previous iteration, the spatial matrix A and background $\boldsymbol{b}$ can be updated by solving the following convex program

There are many solution, so we focus on obtaining the "sparsest" solution. ie. the one with fewest events

$$\begin{aligned}&\underset{A}{\text{minimize}}\ \|A\|_1,\\ &\text{subject to: } A, \boldsymbol{b} \geq 0, \quad \|Y(i,:) - A(i,:)C - b(i)\boldsymbol{f}^\top\| \leq \sigma_i\sqrt{T},\ i = 1, 2, \ldots, d.\end{aligned} \tag{P-S}$$

where $A(i,:), Y(i,:)$ denote the i-th rows of A and Y respectively. Although the matrix A is of very large size, $d \times T$, the problem (P-S) can be readily parallelized into d programs for each pixel separately. Each of these problems can be solved either by using the non-negative LARS algorithm or by using a dual ascent method. LARS is preferred here, since the number of neurons that overlap in a given pixel (and therefore the dimension of the resulting LARS problem) is in general very small.

When the fluorescence from each neuron is highly localized near the soma, the process of estimating A at the k-th iteration can be further sped up by using the previous estimate A^{k-1} as follows. After computing A^{k-1}, we can approximate the center and the size of each neuron (expressed by a column of A), by computing its center of mass and the variance around this center. Then the support of this cell's ROI can be computed as an ellipse centered at the center of mass and rotated along the two principal components that capture most of the variance of the mass of the neuron. When estimating the i-th row of A^k, we can restrict our search to the neurons (columns of A) whose ROIs include the pixel i. This sparsens A^k significantly and makes the dimensionality of each LARS subproblem much smaller, leading to a highly efficient and parallelizable update.

Note that we have not yet incorporated any prior information about the detailed shape of the spatial components $A(i,:)$, which enabled the highly parallel approach described above. However, in many cases it is natural to assume that $A(i,:)$ is connected, or smooth in a suitable sense. Empirically we have found it helpful to include a mild post-processing step at each iteration, using standard non-linear image filtering techniques, such as median filtering or morphological opening, which are effective in removing isolated pixels that appear as active. The removed pixels can then be absorbed by the background component.

Estimating $C, \boldsymbol{f}$: For the temporal components we want to introduce a sparsity penalty to the spiking signal of each neuron to prevent overfitting. We can again use our estimates of the noise variance as hard

[2] Higher rank terms can also be used here if necessary.

Given convex, differentiable fL R^n -> R, if we are at a point x such that f(x) is minimized along each coordinate axis, have we found a global minimizer? ie foes f(x +d.ei) >= f(x) for all d, i => f(x) = min(z) f(z)?

Here ei is the ith standard basis

constraints and derive a parameter-free convex program:

$$\begin{aligned} &\underset{\boldsymbol{c}_1,\dots,\boldsymbol{c}_K,\boldsymbol{f}}{\text{minimize}} \ \sum_{j=1}^{K} \mathbf{1}^\top G\boldsymbol{c}_j, \\ &\text{subject to: } \ G\boldsymbol{c}_j \geq 0, \ j = 1,2,\dots,K \\ &\qquad\qquad \|Y(i,:) - A(i,:)C - b(i)\boldsymbol{f}^\top\| \leq \sigma_i\sqrt{T}, \ i = 1,2,\dots,d. \end{aligned} \tag{P-T}$$

Since the constraints $G\boldsymbol{c}_i \geq 0$ couple the entries within each row of C, and the residual constraints within each column, the program (P-T) cannot be readily parallelized. Moreover, the large number of constraints and the potentially large number of neurons K make the direct solution of (P-T) computationally expensive. To overcome this we employ a block-coordinate descent approach where we sequentially update the temporal component $\boldsymbol{c}_j$ of each neuron. If we denote by $A_{:\setminus j}$ ($C_{\setminus j:}$) the matrix A (C) with its j-th column (row) removed, and $F_{\setminus j} = A_{:\setminus j}C_{\setminus j:} + \boldsymbol{b}\boldsymbol{f}^\top$, then each $\boldsymbol{c}_j$ can be found by solving

$$\begin{aligned} &\underset{\boldsymbol{c}_j}{\text{minimize}} \ \mathbf{1}^\top G\boldsymbol{c}_j, \\ &\text{subject to: } \ G\boldsymbol{c}_j \geq 0, \quad \|Y(i,:) - F_{\setminus j}(i,:) - a_j(i)\boldsymbol{c}_j^\top\| \leq \sigma_i\sqrt{T}, \ i = 1,2,\dots,d. \end{aligned} \tag{7}$$

To solve (7) we employ a dual ascent method where we introduce a set of Lagrange multipliers $\boldsymbol{\lambda}_j \in \mathbb{R}^d$ and consider the function

$$\mathcal{L}(\boldsymbol{c}_j, \boldsymbol{\lambda}_j) = \begin{cases} \mathbf{1}^\top G\boldsymbol{c}_j + \sum_{i=1}^{d} \lambda_j(i)(\|Y(i,:) - F_{\setminus j}(i,:) - a_j(i)\boldsymbol{c}_j^\top\|^2 - \sigma_i^2 T), & G\boldsymbol{c}_j \geq 0 \\ \infty, & \text{otherwise} \end{cases}$$

For a given value of $\boldsymbol{\lambda}_j$, $\mathcal{L}(\boldsymbol{c}_j, \boldsymbol{\lambda}_j)$ can be easily minimized with respect to $\boldsymbol{c}_j$ in $O(T)$ time using the FOOPSI log-barrier interior point method. Then the vector $\boldsymbol{\lambda}_j$ can be updated as

$$\boldsymbol{\lambda}_j^k = \boldsymbol{\lambda}_j^{k-1} - \alpha_k \nabla_{\boldsymbol{\lambda}_j} \mathcal{L}(\boldsymbol{c}_j^k, \boldsymbol{\lambda}_j).$$

Note that at iteration k, $\boldsymbol{\lambda}_j$ can be warm started with the value $\boldsymbol{\lambda}_j^{k-1}$. Moreover, to increase speed we can consider only a small number of the hard constraints, e.g., corresponding to the pixels where the spatial component $\boldsymbol{a}_j$ has the highest values. After we update all the temporal components we also update the background activity vector $\boldsymbol{f}$ by solving a simple nonnegative least squares program. (Note that we do not enforce a sparsity constraint on either $\boldsymbol{f}$ or $\boldsymbol{b}$.)

This block-coordinate descent approach provides a tractable update for the estimates of the temporal component C. Although this method is in general not parallel it can be parallelized to a large extent by exploiting the fact that the different ROIs as specified through the columns of the matrix A do not overlap significantly. At every iteration we can form a graph where each vertex corresponds to a neuron and two neurons are connected with an edge if their spatial filters overlap. Then we can update the temporal components as follows: First, we find the minimum set of vertices such that when removed, the graph is split into a set of disjoint subgraphs. This problem is known as the minimum vertex cover problem in graph theory. It is known to be NP-complete, although for sparse graphs certain approximation algorithms perform well in practice (Vazirani, 2001). After we find this set, we update the temporal components of these neurons, and then we repeat this process in each of the subgraphs in parallel. We can repeat this temporal block-coordinate descent approach until convergence, which typically occurs only after a few iterations.

Merging existing ROIs: Depending on the initialization procedure, a neuron can sometimes be initially split into two or more different ROIs, that subsequently need to be merged. To detect ROIs that need to be merged, we again construct a graph where each vertex corresponds to a neuron and two neurons are connected with an edge if their ROIs overlap. For this graph we detect all the maximal cliques, i.e., the cliques of the

Algorithm 1 Constrained matrix factorization for spatiotemporal spike inference

```
Initialize $A$, $C$, $\boldsymbol{b}$, $\boldsymbol{f}$ using the greedy initialization approach.
repeat
    Compute center of mass for each spatial component and define search region for each pixel.
    Update spatial components $A$ and background component $\boldsymbol{b}$ by solving (P-S).
    Perform median filtering for each component. Absorb discarded pixels into the background.
    Update temporal components $C$ and background activity $\boldsymbol{f}$ (P-T) using coordinate descent.
    Merge overlapping spatial components with overly highly correlated temporal components.
    Remove overly weak components.
until convergence
Order components and let user choose cutoff value $\hat{K}$.
```

graph that are not part of larger cliques. This is again an NP-complete problem which can nevertheless be solved efficiently for large sparse graphs (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 ROIs. A similar merging procedure is also performed when a component is significantly correlated with the background activity, in which case the component is absorbed into the background.

Removing ROIs: After each iteration we can remove any ROIs that do not contribute significantly to the overall spatiotemporal activity, by discarding ROIs that are spatially nearly empty and/or do not contribute any spiking activity. After the last iteration, we normalize each spatial component to have unit energy, and then sort the components based on the maxima of their temporal factors. Empirically we find that this approach effectively orders the neurons, with the first components in the list having strong but sparse temporal activity, making it easy for the user to set a cutoff $\hat{K}$ above which the obtained activity is retained, and below which the components are discarded.

3.2 Greedy initialization of the matrix factorization approach

The matrix factorization approach presented in section 3.1 allows us to effectively separate the different neurons and deconvolve their spikes in a computationally tractable way. However, this approach is bi-convex and can converge to a local maximum point that depends on the specific initialization of the matrix factorization procedure. We designed a custom greedy method for approaching this problem in a computationally efficient way. At every iteration the spatiotemporal data matrix is spatially filtered with a Gaussian kernel of width similar to the size of the neuron. The algorithm finds the location where this filtering procedure explains the maximum variance and draws a square ROI of size roughly twice the size of an average neuron. Within this ROI a rank 1 nonnegative matrix factorization (initialized with the rank-1 SVD of this small patch) is performed to initialize the spatial and temporal components, and the product of these components is then subtracted from the observed data. This procedure is repeated until a user specified number of neurons is located. Then, the resulting residual signal is used to initialize the background component $B = \boldsymbol{b}\boldsymbol{f}^\top$ using rank-1 nonnegative matrix factorization. As discussed above, the total number of neurons is unknown and in practice may be difficult to estimate automatically. A natural strategy is to begin with an overly large value of K, subsequently merging or removing components after each constrained matrix factorization step, or the weakest neurons (defined by the ordering described above) can be discarded in the final step of the algorithm, with very modest user input (a simple choice of the cutoff value $\hat{K}$). A full description of this initialization procedure is given in the supplement. Our full procedure for spike inference and component demixing is schematically represented as Algorithm 1.

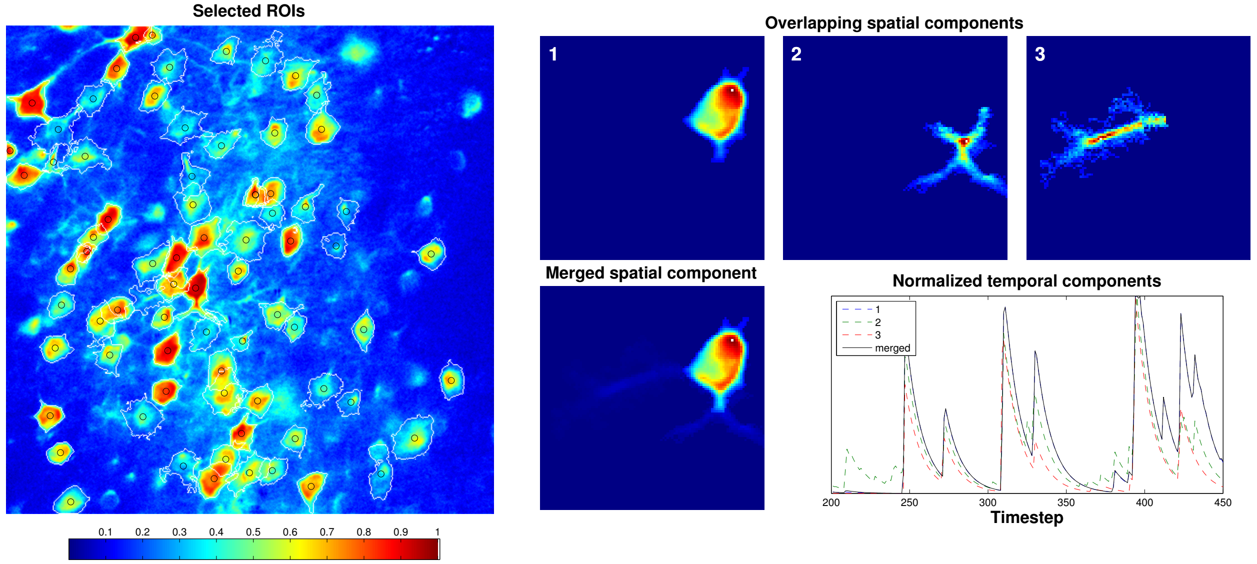

Figure 1: Application to mouse V1 *in vivo* data. Left: Inferred ROIs superimposed on the correlation image of the raw data. The white contours show the boundary of each inferred cell shape; black circles denote the corresponding centers of mass. The supplementary video provides a more complete depiction of the results. Right: Depiction of the merging operation: Upper panels: Three overlapping spatial ROIs with highly correlated temporal components. Lower left: Merged spatial component. Lower right: Estimated temporal components of individual (dashed) and merged (solid) components.

4 Results

Application to population imaging data: We begin by applying our methods to *in vivo* mouse V1 spontaneous activity data. The results are shown in Fig. 1. The algorithm was initialized with the procedure described above with 85 components. During the factorization iterations 10 components were eliminated due to merging operations or negligible total contribution. The contours of the final 75 ROIs are depicted in the left panel of Fig. 1, superimposed on the "correlation image" (Smith and Häusser, 2010) of the raw data. The correlation image for each pixel is computed by averaging the correlation coefficients (taken over time) of each pixel with its 8 immediate neighbors. Localized regions in the correlated image with high intensity correspond to strongly active cells, whereas localized regions with lower intensity correspond to neurons with lower intensity or other non-stationary processes. The algorithm efficiently identifies neurons with very few visually-apparent false positives. The results are viewed best in the video included in the supplementary material. The remaining panels display an example of the merging procedure. The neuron depicted in the lower panel is initially split across three components (upper panels; one for the soma and two for identified dendrites). Since the temporal activity of these components is highly correlated they are merged into a single cell.

Fig. 2 highlights the importance of the demixing procedure. The two identified neurons in the left panel overlap over the region displayed in brown. When analyzing neuron 1, averaging over all of its spatial mask (without first excluding the activity of neuron 2) introduces false spikes (as depicted in the blue trace of the upper right panel) since the activity of neuron 2 is much stronger and influences the analysis even when a relatively small overlap is present.

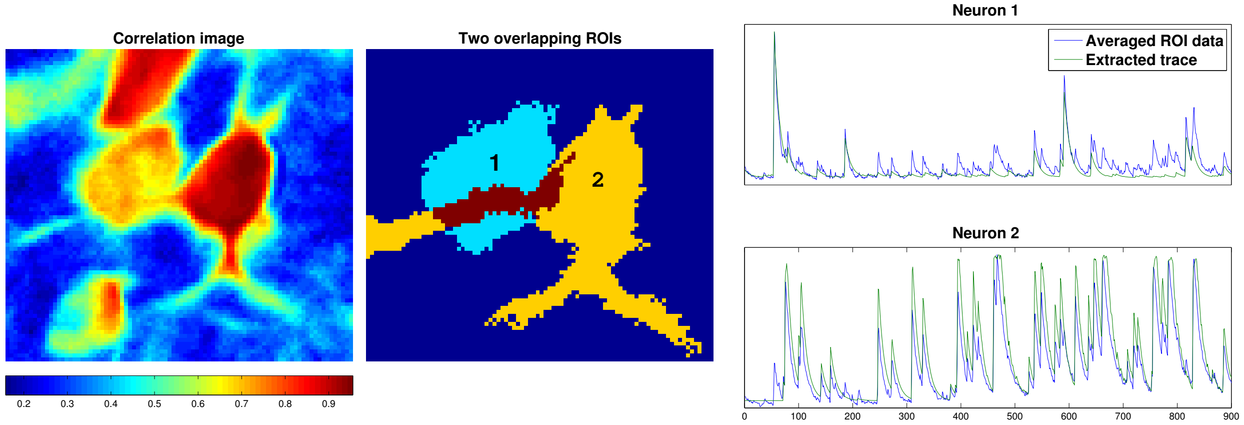

Figure 2: Importance of the demixing procedure. Left: Correlation image zoomed into two overlapping neurons. The exact spatial masks cannot be readily resolved. Middle: Spatial masks for two identified overlapping ROIs after demixing (the brown region corresponds to the overlapping pixels). Right: Inferred calcium traces of the two neurons obtained by averaging over the spatial masks (blue traces) and by applying the proposed method (green traces). Simple averaging over the ROI of neuron 1 can be misleading if demixing is not performed.

Application to dendritic imaging data: A key advantage of the proposed structured matrix factorization framework is that we can apply similar methods to dendritic imaging data where the imaging focuses on the dendrites of multiple neurons and not on the cell bodies. In this case each spatial component corresponds to a set of dendritic branches from a given neuron and the temporal component to the synchronous activity of these branches. The goal is to segment these movies and disentangle the various dendritic branches. These images exhibit certain qualitative differences compared to somatic imaging. Each spatial component is again sparse but is no longer spatially localized since dendritic branches can stretch significantly along the observed imaging plane. As a result, the degree of overlap between the different branches is significantly higher, making even rough interpretation by eye a challenging task; the correlation image in this setting provides very little useful segmentation information. Moreover, the bound calcium dynamics no longer follow somatic calcium indicator dynamics since they are affected by the highly nonlinear processing that takes place in the dendrites. Dropping the temporal dynamics and spatial localization constraints from our problem, we obtain a simpler sparse nonnegative matrix factorization problem which can still be solved efficiently using the methods described above.

We applied this approach to *in vivo* dendritic imaging data taken from the apical dendrites of layer-5 pyramidal neurons in the rodent barrel cortex and show the results in Fig. 3. The raw data is typically dense both in time and space as can be seen from the low intensity of the raw data correlation image (upper left panel). We initialize using a large number of components (in this case 50), and then select the components that correspond to actual dendritic structure using the sorting procedure described above. The resulting denoised movie displays much stronger local correlations (Fig. 3 second column), with the top 20-30 inferred components corresponding to localized dendritic structures that are sparsely active at specific points in time. Four of these components are shown in Fig. 3. A more informative movie of the results as well as a depiction of the sorted ROIs and temporal components can be found in the supplementary material.

To conclude, these results demonstrate that the proposed matrix factorization methods provide powerful tools for analyzing large scale calcium imaging datasets and extracting informative spatiotemporal components.

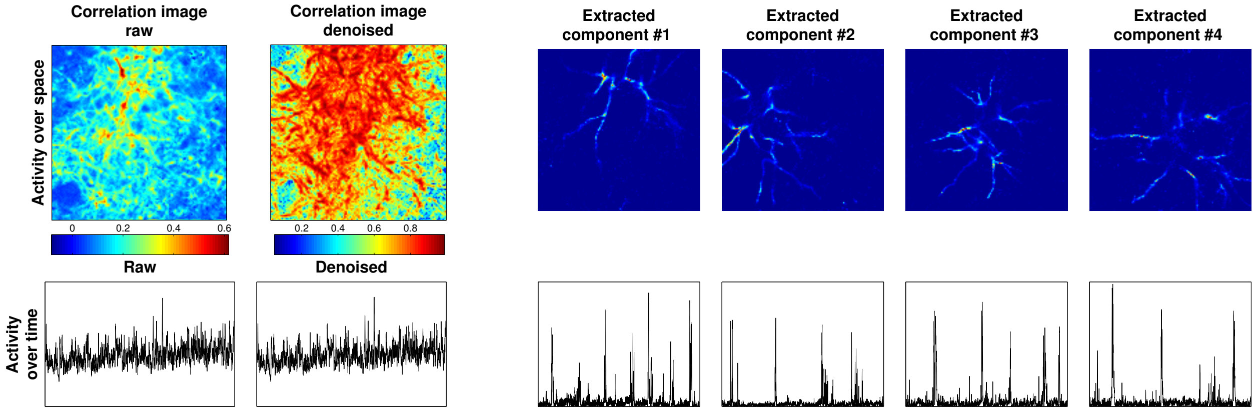

Figure 3: Application to *in vivo* dendritic imaging data from rodent barrel cortex. Top: Correlation images of the raw and denoised data, and four extracted components. Bottom: Spatially averaged activity over time of the raw and denoised data and the temporal traces of the 4 extracted components. The proposed method can segment the dense dendritic imaging data and reveal a rich underlying sparse structure. The supplementary video provides a more complete depiction of the results.

A Algorithms for solving the single pixel constrained deconvolution problem

We briefly discuss the three different approaches that can be used to solve program (P-1d), which we repeat here in an equivalent form for completeness:

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

Dual ascent methods: We introduce Lagrange multipliers for the constraints and define as $\boldsymbol{c}^\lambda, c_1^\lambda$ as the solution to the following program

$$\begin{aligned} \underset{\boldsymbol{c}, c_1}{\text{minimize}} \quad & \mathcal{L}(\boldsymbol{c}, \boldsymbol{c}_{\text{in}}, \lambda) = \mathbf{1}^T G(\boldsymbol{c} - \boldsymbol{c}_{\text{in}}) + \lambda(\|\boldsymbol{y} - \boldsymbol{c} - b\mathbf{1}_T\|^2 - \sigma^2 T), \\ \text{subject to:} \quad & G(\boldsymbol{c} - \boldsymbol{c}_{\text{in}}) \geq 0 \quad c_1 \geq 0. \end{aligned} \tag{8}$$

The problem (8) can be readily solved in $O(T)$ time with the interior point method of Vogelstein et al. (2010). After solving (8), the Lagrange multiplier can be updated as

$$\lambda_k = \lambda_{k-1} - a_k \nabla_\lambda \mathcal{L}(\boldsymbol{c}^{\lambda_{k-1}}, \boldsymbol{c}_{\text{in}}^{\lambda_{k-1}}, \lambda) = \lambda_{k-1} - a_k(\|\boldsymbol{y} - \boldsymbol{c} - b\mathbf{1}_T\|^2 - \sigma^2 T), \tag{9}$$

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

Conic programming: The program of (P-1d) can also be solved with standard interior point methods for conic programming. Due to the simplicity of the residual and non-negativity constraints the solution can be efficiently computed in $O(T)$ 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

$$\begin{aligned}
&\underset{\boldsymbol{s}}{\text{minimize}} && \frac{1}{2\sigma^2}\|\boldsymbol{y} - G^{-1}\boldsymbol{s} - b\mathbf{1}_T\|^2 + \lambda\mathbf{1}^T\boldsymbol{s},\\
&\text{subject to:} && \boldsymbol{s} \geq 0\\
& && \|\boldsymbol{y} - G^{-1}\boldsymbol{s} - b\mathbf{1}_T\| \leq \sigma\sqrt{T}.
\end{aligned} \tag{10}$$

The solution path is computed in the standard piecewise linear way starting from $\lambda_0 = \infty$. As λ decreases, more spikes are added in the solution reducing the energy of the residual signal. The path algorithm is stopped when the produced solution satisfies the residual constraint with equality. Let λ_{k-1}, λ_k be the values of λ at the $(k-1)$-th and k-th step of the algorithm respectively, and $\boldsymbol{s}^{k-1}$, $\boldsymbol{s}^k$ the corresponding solutions. k is chosen such that

$$\|\boldsymbol{y} - G^{-1}\boldsymbol{s}^k - b\mathbf{1}_T\| \leq \sigma\sqrt{T} \leq \|\boldsymbol{y} - G^{-1}\boldsymbol{s}^{k-1} - b\mathbf{1}_T\|.$$

Between the $(k-1)$-th and k-th steps the solution changes according to the direction

$$\boldsymbol{s}_{\text{dir}} = -(\boldsymbol{s}^k - \boldsymbol{s}^{k-1})/(\lambda^k - \lambda^{k-1}),$$

and the solution $\boldsymbol{s}^*$ can be found by finding the solution λ^* of the quadratic equation

$$\|\boldsymbol{y} - \lambda G^{-1}(\boldsymbol{s}^{k-1} + \boldsymbol{s}_{\text{dir}}) - b\mathbf{1}_T\|^2 = \sigma^2 T,$$

and setting

$$\boldsymbol{s}^* = \boldsymbol{s}^{k-1} + (\lambda^{k-1} - \lambda^*)\boldsymbol{s}_{\text{dir}}. \tag{11}$$

Note that this approach does not differentiate between the initial concentration $\boldsymbol{c}_{\text{in}}$ and the spiking signal $\boldsymbol{s}$. However this difference does not affect the solution significantly beyond the first timestep.

The LARS approach is particularly efficient when the spiking signal is expected to be very sparse so the algorithm stops only after a few steps. For more dense spiking, the LARS algorithm can require more steps to achieve the constraints and thus it can be inefficient. However, the LARS algorithm can produce a solution even when the problem (P-1d) is infeasible. In this case, the LARS algorithm computes the full path and produces a (dense) solution for $\lambda = 0$ that satisfies the nonnegativity constraints.

B Captions for supplementary videos and supplementary figure

Video S1: `http://www.stat.columbia.edu/~eftychios/movies/Kira-vid2.mp4`
Application to GCaMP6s-expressing neurons in cortical layer 2/3 of adult mouse V1. Expression was achieved via viral injection of AAV1-hsyn-GCaMP6s into C57Bl/6 mice, three weeks prior to imaging. Frame-scanned, two-photon imaging (950nm excitation wavelength, 535/50 emission filter) was carried out at 3Hz using a 25x (1.05 N.A.) objective. No motion correction was done on the data. Top row: Left: Raw data, Middle: Denoised data without synchronized background activity, Right: Residual signal at 4× 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 algorithm successfully denoises the signal and demixes the overlapping neurons.

Video S2: `http://www.stat.columbia.edu/~eftychios/movies/clay-denp-color.mp4`
Application to 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. Resulting TIF stacks were motion corrected with a dynamic programming algorithm presented in Kaifosh et al. (2013). Top row: Left: Raw data, Middle: Denoised data with the background and noisy components removed. Right: Residual signal at 2× finer scale. Bottom panels: 7 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.

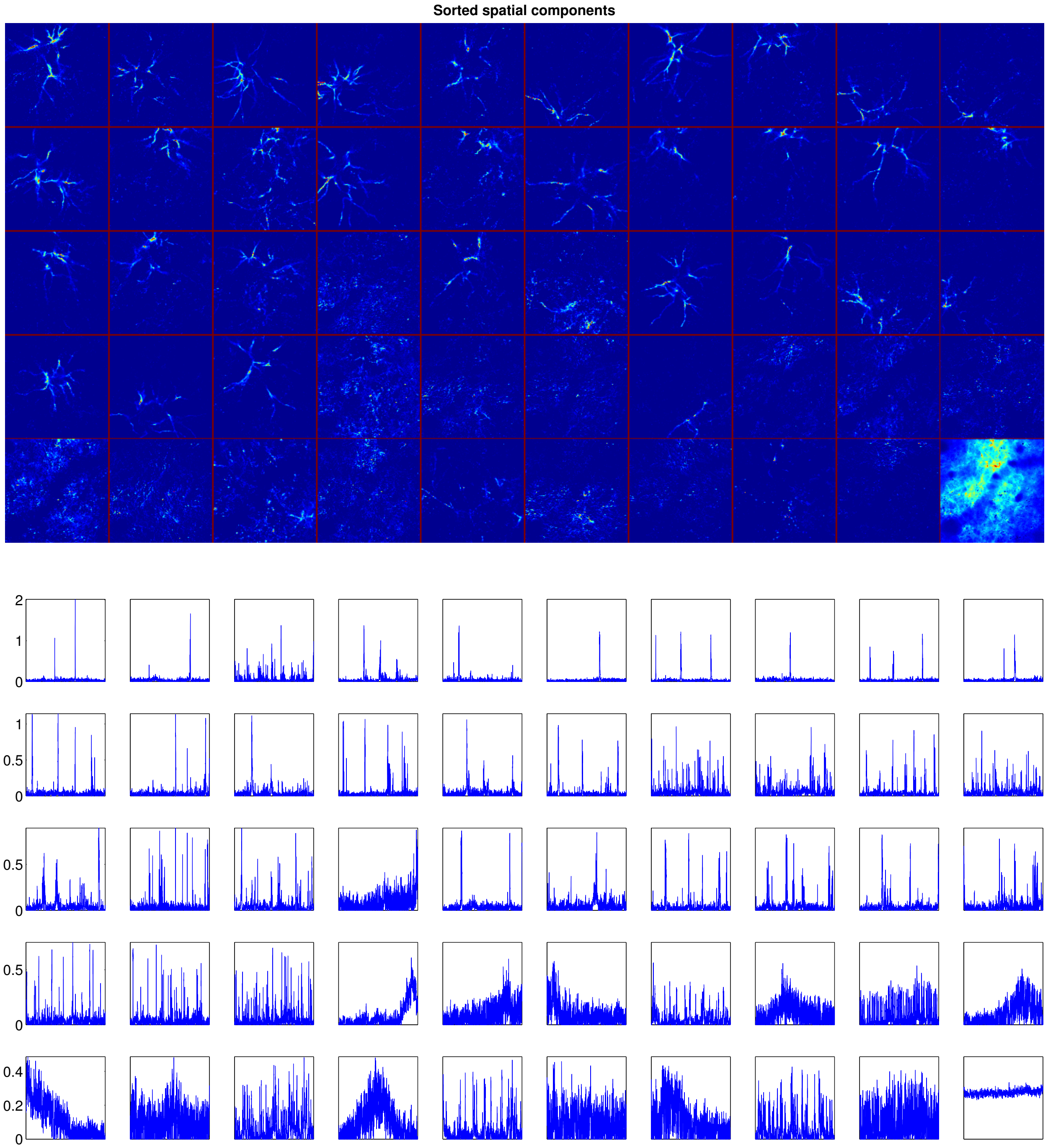

Figure 4: Sorting of the spatial components (top) based on the maximum of their temporal activity (bottom). The structured dendritic extracted components are typically sorted before the noisy components because of their activity is sparse in time but with high magnitude, indicating actual dendritic activity.

The convolution of two 2D functions f and g is defined as the volume of product of f and "shifted" g.
The second function g is sometimes called "weight". since it determines, how much of 'f' will get into the result.

C Algorithmic description for the greedy initialization procedure

Algorithm 2 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 to scan $\tau = (\tau_x, \tau_y)$; window size $w = (w_x, w_y)$.
  procedure GreedyNeuronId($Y$, $K$, $\tau$, $w$)
      $R = Y$;
      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 $(\tau_x^2, \tau_y^2)$, 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, $\rho = D^T R$, $v_i = \sum_{t=1}^{T} \rho_{it}$;
          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

              $\underset{\boldsymbol{a}_k \in \mathbb{R}^d, \boldsymbol{c}_k \in \mathbb{R}^T}{\text{minimize}} \quad \|R - \boldsymbol{a}_k \boldsymbol{c}_k^T\|^2$
              $\text{subject to:} \quad a_k(i) \ge 0, i \in S_k$
              $\qquad\qquad\quad a_k(i) = 0, i \notin S_k.$                                        (12)

          $R \leftarrow R - \boldsymbol{a}_k \boldsymbol{c}_k^T$;
      end for
      $R \leftarrow R + \boldsymbol{m}\mathbf{1}_T^T$. Add median values back to the residual and solve

              $\underset{\boldsymbol{b} \in \mathbb{R}^d, \boldsymbol{f} \in \mathbb{R}^T}{\text{minimize}} \quad \|R - \boldsymbol{b}\boldsymbol{f}^T\|^2$
              $\text{subject to} \quad b(i) \ge 0, \ i = 1, \ldots, d$                              (13)
              $\qquad\qquad\quad f(t) \ge 0, \ t = 1, \ldots, T.$

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

Gaussian blur of a 2D function is defined as the convolution of that function with 2D Gaussian function.

At the beginning we center the data at each pixel around zero by subtracting the median over time. At each iteration, we use a (truncated) Gaussian kernel of size similar to a neuron to scan the residual and identify the location where the kernel explains the most variance over time. Then a rank-1 matrix factorization extracts the spatial component that is localized around the identified location. After neuron k has been identified, the inferred signal is subtracted to update the residual for the next neuron. When all K neurons have been identified, the median is added back to the residual signal and the background activity is estimated using by solving a rank-1 nonnegative matrix factorization problem.

Note that since the spatial component is localized, at each step only a small portion of the residual is updated and therefore only a small portion of the explained variance v_i needs updating. Also note that the matrix factorization step (equation (12) in algorithm 2) 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 5 iterations are enough for convergence. solution of equation (12) is non-identifiable by a scalar multiplication, and we can simply identify the result by constraining spatial components to have unit norm.

References

Ahrens, M. B., M. B. Orger, D. N. Robson, J. M. Li, and P. J. Keller (2013). Whole-brain functional imaging at cellular resolution using light-sheet microscopy. Nature methods 10(5), 413–420.

Akaike, H. (1969). Fitting autoregressive models for prediction. Annals of the institute of Statistical Mathematics 21(1), 243–247.

Andilla, F. D. and F. A. Hamprecht (2013). Learning multi-level sparse representations. In Advances in Neural Information Processing Systems, pp. 818–826.

Chen, T.-W., T. J. Wardill, Y. Sun, S. R. Pulver, S. L. Renninger, A. Baohan, E. R. Schreiter, R. A. Kerr, M. B. Orger, V. Jayaraman, et al. (2013). Ultrasensitive fluorescent proteins for imaging neuronal activity. Nature 499(7458), 295–300.

Efron, B., T. Hastie, I. Johnstone, R. Tibshirani, et al. (2004). Least angle regression. The Annals of statistics 32(2), 407–499.

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

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

Grewe, B. F., D. Langer, H. Kasper, B. M. Kampa, and F. Helmchen (2010). High-speed in vivo calcium imaging reveals neuronal network activity with near-millisecond precision. Nature methods 7(5), 399–405.

Haeffele, B., E. Young, and R. Vidal (2014). Structured low-rank matrix factorization: Optimality, algorithm, and applications to image processing. In Proceedings of the 31st International Conference on Machine Learning (ICML-14), pp. 2007–2015.

Kaifosh, P., M. Lovett-Barron, G. F. Turi, T. R. Reardon, and A. Losonczy (2013). Septo-hippocampal GABAergic signaling across multiple modalities in awake mice. Nature neuroscience 16(9), 1182–1184.

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

Mukamel, E. A., A. Nimmerjahn, and M. J. Schnitzer (2009). Automated analysis of cellular signals from large-scale calcium imaging data. Neuron 63(6), 747–760.

Oñativia, J., S. R. Schultz, and P. L. Dragotti (2013). A finite rate of innovation algorithm for fast and accurate spike detection from two-photon calcium imaging. Journal of neural engineering 10(4), 046017.

Pachitariu, M., A. M. Packer, N. Pettit, H. Dalgleish, M. Hausser, and M. Sahani (2013). Extracting regions of interest from biological images with convolutional sparse block coding. In Advances in Neural Information Processing Systems, pp. 1745–1753.

Pnevmatikakis, E., T. Machado, L. Grosenick, B. Poole, J. Vogelstein, and L. Paninski (2013). Rank-penalized nonnegative spatiotemporal deconvolution and demixing of calcium imaging data. In Computational and Systems Neuroscience Meeting COSYNE.

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

Pnevmatikakis, E. and L. Paninski (2013). Sparse nonnegative deconvolution for compressive calcium imaging: algorithms and phase transitions. In Advances in Neural Information Processing Systems, Volume 26, pp. 1250–1258.

Prevedel, R., Y.-G. Yoon, M. Hoffmann, N. Pak, G. Wetzstein, S. Kato, T. Schrödel, R. Raskar, M. Zimmer, E. S. Boyden, and A. Vaziri (2014). Simultaneous whole-animal 3*d*-imaging of neuronal activity using light field microscopy. Nature Methods. Advance online publication, doi:10.1038/nmeth.2964.

Smith, S. L. and M. Häusser (2010). Parallel processing of visual space by neighboring neurons in mouse visual cortex. Nature neuroscience 13(9), 1144–1149.

Vazirani, V. V. (2001). Approximation algorithms. Springer.

Vogelstein, J., A. Packer, T. Machado, T. Sippy, B. Babadi, R. Yuste, and L. Paninski (2010). Fast non-negative deconvolution for spike train inference from population calcium imaging. Journal of Neurophysiology 104(6), 3691–3704.

Vogelstein, J., B. Watson, A. Packer, R. Yuste, B. Jedynak, and L. Paninski (2009). Spike inference from calcium imaging using sequential monte carlo methods. Biophysical journal 97(2), 636–655.